Universidade Federal
do Rio de Janeiro
Escola Politécnica

ESTUDO DE ESTRATÉGIAS DE CONTROLE DE ESTOQUE COM OTIMIZAÇÃO VIA ALGORITMOS GENÉTICOS

ANDRÉ MILHORANCE DE CASTRO

TIAGO SALVIANO CALMON

Projeto de Graduação apresentado ao Curso de Engenharia de Controle e Automação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientadores:
Prof. Amit Bhaya, Ph.D
Prof. Eugenius Kaszkurewicz, D.Sc

Rio de Janeiro, RJ - Brasil

2013

ESTUDO DE ESTRATÉGIAS DE CONTROLE DE ESTOQUE COM OTIMIZAÇÃO VIA ALGORITMOS GENÉTICOS

ANDRÉ MILHORANCE DE CASTRO

TIAGO SALVIANO CALMON

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE CONTROLE E AUTOMAÇÃO

Aprovado por:

Prof. Amit Bhaya, Ph.D

Prof. Eugenius Kaszkurewicz, D.Sc

Prof. Alessandro Jacoud Peixoto, D.Sc

Rio de Janeiro, RJ - Brasil

2013

Castro, André Milhorance. Calmon, Tiago Salviano.
Estudo de Estratégias de Controle de Estoque com Otimização via Algoritmos Genéticos
106 páginas
Projeto de Graduação UFRJ/ Escola Politécnica/Curso de Engenharia de Controle e Automação, 2013.
Orientadores:
Prof. Amit Bhaya, Ph.D
Prof. Eugenius Kaszkurewicz, D.Sc

1. Teoria de Controle
2. Controle de Estoque
3. Algoritmos Genéticos
4. Lógica *Fuzzy*

I. Castro, André Milhorance. II Calmon, Tiago Salviano. IV. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia de Controle e Automação. V. Título.

Agradecimentos, por André

Sem dúvidas, é impossível atingir o sucesso, em qualquer grau, apenas com esforços individuais. Estes são fundamentais, mas é preciso reconher o auxílio direto e indireto dos que caminham ao nosso lado. Nada mais justo, então, que agradecer a todos que participaram dessa minha jornada até aqui.

Começo, assim, agradecendo aos meus pais, Carlos Frederico e Maria das Graças, que entendem o valor da educação como uma união de instrução acadêmica e valores sociais.

Agradeço, também, as minhas irmãs pelo exemplo de dedicação.

Reconheço a extrema importância de Lilian Maria, que me ilumina com seu amor, amizade e sabedoria.

Agradeço, ao meu companheiro de trabalho Tiago Calmon, pela sua paciência, inteligência e confiância ao me convidar para o projeto. Sem dúvidas, nossas reuniões foram um exemplo de "produtividade".

Não seria possível realizar um trabalho de qualidade sem a estimada orientação dos professores Amit Bhaya e Eugenius Kaszkurewicz. Muito obrigado aos dois pelos conhecimentos e valiosas discussões e orientações.

Agradeço aos meus familiares, pelo amparo em todos os momentos.

Aos integrantes da maravilhosa turma de Controle-T12. Sem dúvidas, teria sido mais difícil sem vocês. Companherismo sempre.

À equipe da Promon Engenharia, pela compreensão nos momentos necessários e pelo exemplo de excelência.

Por fim, a todos os mestres, queridos professores, com os quais aprendi o valor do conhecimento.

A todos vocês, o meu muito obrigado.

André Milhorance de Castro

Agradecimentos, por Tiago

Em minha jornada nos anos em que fui aluno de graduação da UFRJ muitas pessoas foram, direta ou indiretamente, fundamentais para meu sucesso na difícil tarefa de concluir o curso. Gostaria de agradecer a algumas delas, em especial.

Começo por meu pais, Francisco Calmon e Susana Salviano, que nunca mediram esforços para que eu pudesse dispor da melhor educação possível, além de serem fontes inesgotáveis de motivação.

A meu grande amigo André Calmon, que tenho a sorte de ser meu irmão. A meus irmãos Lucas Calmon e Giovana Salviano que estiveram ao meu lado durante todo esse tempo.

A Suzana Braz, que esteve ao meu lado em todos os bons e maus momentos, que é namorada amorosa, amiga e conselheira. Tenho certeza que não teria sido possível sem você.

Aos meus avós Francisco Carlos Calmon, Laïs Calmon e Izabel Salviano e em memória de meu avô Marinho Salviano, que são exemplos de vida.

Agradeço especialmente a meu companheiro de trabalho André Milhorance, que com sua organização, pró-atividade, inteligência e liderança nos conduziu por todo esse trabalho com maestria. Sua capacidade de trabalho me inspirou no curso da elaboração dessa tese.

A todos os alunos de Controle que estiveram a meu lado, seja para compartilhar tarefas ou boas risadas. Agradecimento especial a todos os integrantes da T-12, que me abraçaram e me receberam como irmão, e aos amigos Marco Fernandes e Arthur Fernandes.

Aos professores Amit Bhaya e Eugenius Kaszkurewicz que sempre foram solícitos em nos orientar e sempre forneceram dicas valiosas. A confiança e presteza de vocês foi de suma importância.

Muito Obrigado!

Tiago Salviano Calmon

Resumo do Projeto de Graduação apresentado à Escola Politécnica/ UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Controle e Automação.

ESTUDO DE ESTRATÉGIAS DE CONTROLE DE ESTOQUE COM OTIMIZAÇÃO VIA ALGORITMOS GENÉTICOS

André Milhorance de Castro

Tiago Salviano Calmon

2013

Orientadores:

Prof. Amit Bhaya, Ph.D

Prof. Eugenius Kaszkurewicz, D.Sc

Curso: Engenharia de Controle e Automação

Uma questão importante para a logística empresarial é a administração de centrais de estoque. Como aumentar a produtividade e, principalmente, o lucro por meio de uma gestão otimizada desse componente chave em diversos seguimentos industrio-comerciais?

Esse texto se dedica ao estudo de uma cadeia de suprimentos simples sujeita a diferentes estratégias de controle sempre visando sua gestão de forma ótima.

O objetivo desse trabalho é a avaliação de diferentes estratégias de controle que otimizem o EVA (*Economic Value Added*) de um sistema de gestão de estoque ao final do período de um ano. Para tanto,as estratégias estudadas são analisadas por meio de simulações realizadas em plataforma MATLAB.

São analisados controladores PID e com Lógica *Fuzzy* . A demanda é estimada a partir de sistemas de suavização exponencial simples e o método de otimização escolhido é o Algoritmo Genético.

Palavras Chaves: Teoria de Controle, Controle de Estoque, Algoritmos Genéticos, Lógica *Fuzzy*

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Engineer.

STUDY OF INVENTORY CONTROL STRATEGIES WITH GENETIC ALGORITHM OPTIMIZATION

André Milhorance de Castro
Tiago Salviano Calmon
2013

Advisors:

Prof. Amit Bhaya, Ph.D

Prof. Eugenius Kaszkurewicz, D.Sc

Course: Control and Automation Engineering

An important issue in enterprise logistics is inventory warehouse management. How to improve productivity and the profit by means of an optimized management of this key component of several industrial sectors? This work will study an inventory system model, subject to some different control strategies. The goal is to evaluate the control strategies when they are used to optimize the EVA (Economic Value Added) at the end of a specified period. Both PID and Fuzzy Logic Controllers are analyzed. The demand is predicted using simple exponential smoothing methods and to optimize EVA, a Genetic Algorithm is implemented.

Keywords: Control Theory, Inventory Control, Genetic Algorithm, Fuzzy Logic

Conteúdo

RESUMO **5**

ABSTRACT **6**

LISTA DE FIGURAS **iv**

LISTA DE TABELAS **x**

LISTA DE ABREVIATURAS **xii**

LISTA DE SÍMBOLOS **xiii**

1 Introdução **1**

1.1 A Logística Moderna 2

1.1.1 Sistema de Tecnologia da Informação 3

1.1.2 Gestão de Estoques 4

1.2 O Modelo de Estoque 6

1.2.1 *Economic Value Added* - EVA 10

1.2.2 Observações Gerais e Revisão da Literatura 14

1.3 Objetivo 17

1.4 Motivação 18

1.5 Conteúdo do Trabalho 18

2 Revisão Teórica **19**

2.1 Revisão de aspectos de Controle e Otimização 19

2.1.1 Características de Controle Discreto 19

2.2 Introdução ao Controle por Lógica *Fuzzy* 21
2.2.1 Projeto de um Controlador Fuzzy 22
2.2.2 Conceitos Fundamentais 23
2.2.3 Razões para utilização da Lógica *Fuzzy* nesse projeto 28
2.3 Introdução aos Algoritmos Genéticos 29
2.3.1 Geração da População Inicial 30
2.3.2 Codificação em um Alfabeto 31
2.3.3 Reprodução 32
2.3.4 Mutação 34
2.3.5 Nova População 34
2.3.6 Critério de Parada 35
2.4 Introdução às Metodologias de Predição 35
2.4.1 Séries Temporais 36
2.4.2 Modelos de Suavização Exponencial 36
2.4.3 Previsão Indireta da Demanda 37

3 Projeto de Controle **39**

3.1 Métodos Computacionais Utilizados 40
3.1.1 iThink © 40
3.1.2 Berkeley Madonna © 42
3.1.3 Simulink - MATLAB © 43
3.1.4 *M-File* - MATLAB © 43
3.1.5 Características do Simulador 44
3.2 Maximização do EVA em Horizonte Finito 46
3.3 Sistemas de Predição 47
3.3.1 Preditor 1 - Filtro de Segunda Ordem 48
3.3.2 Preditor 2 - Preditor de Suavização Exponencial Simples 49
3.4 Estratégias de Controle 50
3.4.1 Controlador 1 - Modelo Hannon/McGarvey 51
3.4.2 Controlador 2 - Modelo Tosetti 52

3.4.3 Controlador 3 - PID Discreto 53
3.4.4 Controlador 4 - Realimentação *Receiving* 54
3.4.5 Controlador 5 - Controlador Fuzzy 55

4 Simulações e Resultados **61**
4.1 Metodologia de Simulação e Obtenção de Resultados 61
4.1.1 Método de Extração do Valor Ótimo 62
4.1.2 Ajustes no Algoritmo 62
4.2 Apresentação dos Resultados 64
4.2.1 Controlador Garvey-Hannon 64
4.2.2 Controlador PID Tosetti 69
4.2.3 Controlador PID Discreto 74
4.2.4 Controlador Realimentação de *Receiving* 80
4.2.5 Controlador *Fuzzy* 85
4.3 Análise dos Resultados 89

5 Conclusão **94**
5.1 Resultados Gerais 94
5.2 Trabalhos Futuros 95

Referências Bibliográficas **96**

A Funcionamento da *Toolbox Optimization Tools* **100**

Lista de Figuras

1.1 Cadeia de Suprimentos Multiestágio 7
1.2 Esquema Armazém-Loja 7
1.3 Arquitetura de estoque 8
1.4 Controlador APIOBPCS básico - [34] 14

2.1 Esquema de um controlador *Fuzzy* [24] 22
2.2 Função de Pertinência para o valor linguístico *baixa* 26
2.3 Diagrama esquemático do funcionamento dos Algoritmos Genéticos. [30] . . . 30
2.4 Seleção por *Roullete Wheel Selection* para cinco indivíduos. [26] 33

3.1 Modelo em fluxo - iThink (retirado do *site* do desenvolvedor [33]) 41
3.2 Tela gráfica - Berkeley Madonna ©(retirado do *site* do desenvolvedor [19]) . . 43
3.3 Diagrama de funcionamento do Simulador 45
3.4 Sinal Original e Sinal Estimado - Preditor 1 49
3.5 Sinal Original e Sinal Estimado - Preditor 2 50
3.6 Estrutura genérica para os controladores estudados 51
3.7 Estrutura do Controlador 1 - Hannon/McGarvey 52
3.8 Estrutura do Controlador 2 - Tosetti 53
3.9 Estrutura do Controlador 3 - PID Discreto 54
3.10 Estrutura do Controlador 4 - Realimentação Receiving 55
3.11 Estrutura do Controlador 5 - *Fuzzy* 56
3.12 Funções de Pertinência para o Erro do Store - $I_{sp} - I(k)$ 57
3.13 Funções de Pertinência para o Erro do Shipped - $W_{sp} - W(k)$ 58
3.14 Funções de Pertinência para o Sinal de Controle $u(k) = O(k)$ 58

3.15 Superfície de Controle 60

4.1 Evolução do Algoritmo Genético para o Controlador Garvey-Hannon com demanda estimada por séries temporais 65
4.2 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais 65
4.3 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais 66
4.4 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais 66
4.5 Evolução diária do valor do EVA para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais 67
4.6 Evolução do Algoritmo Genético para o Controlador Garvey-Hannon com demanda estimada por suavização exponencial 67
4.7 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial 68
4.8 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e vendas $S(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial 68
4.9 Evolução do nível de pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial . . 69
4.10 Evolução do $EVA(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial 69
4.11 Evolução do Algoritmo Genético para o Controlador PID Tosetti com demanda estimada por séries temporais 70
4.12 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais . . 70

4.13 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais 71
4.14 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais 71
4.15 Evolução diária do valor do EVA para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais 72
4.16 Evolução do Algoritmo Genético para o Controlador PID Tosetti com demanda estimada por suavização exponencial 72
4.17 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial 73
4.18 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial 73
4.19 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial 74
4.20 Evolução diária do valor do EVA para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial 74
4.21 Evolução do Algoritmo Genético para o Controlador PID Discreto com demanda estimada por séries temporais 75
4.22 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais . 75
4.23 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais 76
4.24 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais 76

4.25 Evolução diária do valor do EVA para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais 77
4.26 Evolução do Algoritmo Genético para o Controlador PID Discreto com demanda estimada por suavização exponencial 77
4.27 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial 78
4.28 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial 78
4.29 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial 79
4.30 Evolução diária do valor do EVA para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial 79
4.31 Evolução do Algoritmo Genético para o Controlador Realimentação de *Receiving* com demanda estimada por séries temporais 80
4.32 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais 81
4.33 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais 81
4.34 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais 82
4.35 Evolução diária do valor do EVA para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais 82
4.36 Evolução do Algoritmo Genético para o Controlador Realimentação de *Receiving* com demanda estimada por suavização exponencial 83

4.37 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial 84
4.38 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial 84
4.39 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial 85
4.40 Evolução diária do valor do EVA para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial 85
4.41 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais 86
4.42 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais 86
4.43 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais 87
4.44 Evolução diária do valor do EVA para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais 87
4.45 Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial . 88
4.46 Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial 88
4.47 Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial 89
4.48 Evolução diária do valor do EVA para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial 89

4.49 Evolução diária do valor do EVA para $I_{sp} = 120$, $W_{sp} = 80$, $K_{p1} = 0.1$ e $K_p2 = 4$ 90

4.50 Gráfico comparativo da evolução do EVA para todos os controladores utilizando Preditor 1 92

A.1 Tela da Interface Gráfica da ferramenta *Optimization Tool* 101

Lista de Tabelas

3.1 Regras de Inferência 59

4.1 Parâmetros obtidos pelo GA em dez simulações para o controlador Garvey-Hannon com demanda estimada por séries temporais 64

4.2 Parâmetros e resultados para o controlador Garvey-Hannon com demanda estimada por séries temporais 65

4.3 Parâmetros obtidos pelo GA em dez simulações para o controlador Garvey-Hannon com demanda estimada por suavização exponencial 67

4.4 Parâmetros e resultados para o controlador Garvey-Hannon com demanda estimada por suavização exponencial 68

4.5 Parâmetros obtidos pelo GA em dez simulações para o controlador PID Tosetti com demanda estimada por séries temporais 69

4.6 Parâmetros e resultados para o controlador PID Tosetti com demanda estimada por séries temporais 70

4.7 Parâmetros obtidos pelo GA em dez simulações para o controlador PID Tosetti com demanda estimada por suavização exponencial 72

4.8 Parâmetros e resultados para o controlador PID Tosetti com demanda estimada por suavização exponencial 73

4.9 Parâmetros obtidos pelo GA em dez simulações para o controlador PID Discreto com demanda estimada por séries temporais 74

4.10 Parâmetros e resultados para o controlador PID Discreto com demanda estimada por séries temporais 75

4.11 Parâmetros obtidos pelo GA em dez simulações para o controlador PID Discreto com demanda estimada por suavização exponencial 77

4.12 Parâmetros e resultados para o controlador PID Discreto com demanda estimada por suavização exponencial 78
4.13 Parâmetros obtidos pelo GA em dez simulações para o controlador Realimentação de *Receiving* com demanda estimada por séries temporais 80
4.14 Parâmetros e resultados para o controlador Realimentação de *Receiving* com demanda estimada por séries temporais 80
4.15 Parâmetros obtidos pelo GA em dez simulações para o controlador Realimentação de *Receiving* com demanda estimada por suavização exponencial 83
4.16 Parâmetros e resultados para o controlador Realimentação de *Receiving* com demanda estimada por suavização exponencial 83
4.17 Parâmetros e resultados para o controlador *Fuzzy* com demanda estimada por séries temporais 86
4.18 Parâmetros e resultados para o controlador *Fuzzy* com demanda estimada por suavização exponencial 87
4.19 Melhores valores obtidos pelos controladores ótimos, médias e desvios padrão . 92

Lista de Abreviaturas

3E - Eficiência, eficácia e efetividade

APIOBPCS - Automatic Pipeline Inventory and Order Based Production Control Systems

AR - Modelos Autoregressivos

CRM - Customer Relationship Management

ECR - Efficient Consumer Response

EDI - Electronic Data Interchange

ERP - Entreprise Resources Planning

EVA - Economic Value Added

GA - Genetic Algorithm

JIT - Just in Time

MA - Modelos de Médias Móveis

MIMO - Múltiplas Entradas e Múltiplas Saídas - Multiple Inputs, Multiple Outputs

MISO - Múltiplas Entradas e Saída Única - Multiple Inputis, Single Output

MRP I e II - Material Requirements Planning

PID - Proporcional, Integral e Derivativo

RFID - Radio Frequency Identification

ROI - Return of Investiment

SISO -Single Input, Sigle Output

TI - Tecnologia da Informação

TMS - Transportation Management System

WMS - Warehousing Management System

Lista de Símbolos

$I(k) = store(k)$: volume de estoque acumulado no dia k

$W(k) = shipped(k)$: volume de itens em transição no dia k - *Work-in-Progress*

$O(k) = ordering(k)$: volume de itens pedidos no dia k

$O(k - \tau) = receiving(k)$: volume de itens que chegam ao estoque no dia k

$S(k) = selling(k)$: volume de vendas do produto no dia k

$D(k)$: demanda pelo produto no dia k

$\hat{D}(k)$: demanda estimada para o produto no dia k

k: índice temporal que representa o avanço dos dias

τ: diferença entre o dia em que se faz o pedido e o dia da reposição do estoque - *lead time*

α: parâmetro sintonizado no método de previsão por suavização exponencial simples

$\mathbf{x}$: vetor de variáveis de estados

$\mathbf{u}$: vetor de controle

$\mathbf{b}$: vetor de *bits* do Algoritmo Genético

$EVA(k)$: valor do Economic Value Added no dia k

K_i: Ganho Integral do controlador PID

K_p: Ganho Proporcional do controlador PID

K_d: Ganho Derivativo do controlador PID

$K_{p1} = ReactionStore$: Ganho que multiplica o erro entre o estoque e sua referência

$K_{p2} = ReactionShipped$: Ganho que multiplica o erro entre $W(k)$ e sua referência

$K_{p3} = ReactionReceiving$: Ganho que multiplica o erro entre $O(k - \tau)$ e sua referência

$I_{sp} = StoreTarget$: Referência para $I(k)$

$W_{sp} = ShippedTarget$: Referência para $W(k)$

$R_{sp} = ReceivingTarget$: Referência para $O(k - \tau)$

Capítulo 1

Introdução

Competitividade é um tema sempre em voga no mundo corporativo. Ser produtivo, suportado pelo tripé eficiência, eficácia e efetividade (abreviado 3E), não é meramente uma vantagem que uma empresa pode desenvolver em relação a seus concorrentes. É uma questão de sobrevivência em um ambiente altamente complexo e dinâmico.

Os processos organizacionais necessitam ser continuamente remodelados; qualidade e baixo custo são objetivos primários. A inovação, desse modo, não consiste apenas em gerar novas alternativas em solucionar problemas, mas, também, em reestruturar os meios de produção já colocados em prática. O 3E é a base. Uma metodologia de trabalho é adequada se ela pode ser comparada, ou superior, aos considerados exemplos de mercado em tal contexto.

A logística organizacional trabalha justamente na adequação dos processos de trabalho ao contexto contemporâneo ao qual estão inseridos. Esse ramo do conhecimento visa estudar e implementar estratégias que garantam a competitividade da empresa em seu nicho de mercado.

Uma questão importante para a logística empresarial é a administração de centrais de estoque. Como aumentar a produtividade e, principalmente, o lucro por meio de uma gestão otimizada desse recurso chave em diversos seguimentos industrio-comerciais?

Esse texto tentará elucidar questões sobre a dinâmica de um sistema em que o estoque é o centro (fornecida em [12]), assim como possíveis técnicas, embasadas pela teoria de controle, para sua gestão de forma ótima.

1.1 A Logística Moderna

Não seria falacioso pensar que a logística é uma questão presente em organizações humanas desde que o homem passou a produzir mais do que era capaz de consumir. Era, então, necessário solucionar problemas relacionados a como transportar o excesso, como armazená-lo e como gerar a comunicação para que todos, os autorizados, soubessem como encontrá-lo.

Como se encontra em [35], Daskin, em 1985, tinha a seguinte concepção sobre a logística:

A logística pode ser compreendida como o planejamento e a operação dos sistemas físicos, informacionais e gerenciais necessários para que os insumos e produtos vençam condicionantes espaciais e temporais de forma econômica.

Como se percebe na descrição de Daskin, a logística não trata apenas de sistemas físicos, mas, também, de métodos gerenciais e de informações. Por tal razão, hoje é imprescindível relacionar a área de logística com a Tecnologia da Informação (TI). O parelhamento de estratégias entre as duas permite a gestão integrada dos sistemas descritos por Daskin.

A TI trabalhará com a monitoração dos fluxos de processo ao longo da cadeia de produção, desempenhando funções relativas a aquisição de dados, transferências dos mesmos para uma central de processamento, armazenamento de informação e, em alguns casos, auxílio inteligente na tomada de decisão [35].

Assim, a cadeia logística terá seus componentes primários e secundários monitorados e controlados pela arquitetura de TI. Segundo [35], podemos descrever como componentes primários da logística:

- Processamento de pedidos
- Estoques
- Transporte

Ainda em [35], os componentes secundários são:

- Armazenagem
- Manuseio

- Embalagem

Neste trabalho, abordaremos, principalmente, aspectos referentes ao processamento de pedidos e ao estoque.

1.1.1 Sistema de Tecnologia da Informação

O crescimento no uso de tecnologia nas corporações, como consequência da redução dos custos das mesmas, permite ao gestor total observabilidade do sistema sob sua administração. Ainda, é possível se usufruir de técnicas de inteligência computacional para o auxílio a tomada de decisão.

Segundo [22], já é possível observar sistemas de gerenciamento consolidados, baseados principalmente em TI. Pode-se citar:

- WMS - Warehousing Management System
- MRP I e II - Material Requirements Planning
- ERP - Enterprise Resources Planning
- ECR - Efficient Consumer Response
- CRM - Customer Relationship Management
- TMS - Transportation Management System

Essas plataformas de *software* são viáveis devido ao uso de computadores pessoais, EDI (Electronic Data Interchange), leitura ótica, código de barras e, atualmente também, Identificação por Rádio Frequência (RFID - *Radio Frequency Identification*). Tais arquiteturas de *hardware* permitem que a TI monitore os componentes primários e secundários de uma cadeia logística.

Assim, técnicas de controle automático de gestão de estoques, como aborda este trabalho, se enquadram perfeitamente nas tendências de TI para auxílio à gestão de negócios.

1.1.2 Gestão de Estoques

Segundo [22], estoque é a acumulação armazenada de recursos materiais em um sistema de transformação. O material estocado pode ter caráter objetivo como matéria-prima, ferramentas, produtos manufaturados; mas há a possibilidade de entender o acúmulo de informações através de dados como estoque. De qualquer forma, itens estocados variam de acordo com sua complexidade, qualidade e preço.

Em nosso trabalho, entenderemos o estoque como o armazenamento temporário de um determinado produto manufaturado.

Afinal, qual a razão de haver tal acúmulo? Para [22], o estoque ocorre devido a diferença de ritmo entre o fornecimento e a demanda. Existe a intenção, por parte do vendedor do produto, em sempre atender seu cliente. Este, por sua vez, não consome em intervalos determinísticos, sendo, então, necessária a existência de uma parcela de produtos que atenda a requisições imprevistas. Não ter o que oferecer ao cliente é tido como mau desempenho estratégico.

Ao mesmo tempo que permite suavizar o descasamento entre fornecimento e demanda, estocar significa custo. Para armazenar é necessário um local apropriado, segurança, funcionários, energia e outros fatores. Itens mantidos em estoque também podem deteriorar, tornarem-se obsoletos ou perderem-se.

Assim, o acúmulo sem planejamento pode acarretar na falta de competitividade empresarial da organização. Então, para garantir a correta gestão do estoque, cabe ao administrador, segundo [35]:

- Definir o momento correto da compra
- Definir quantidade ideal
- Buscar os melhores preços
- Atender os níveis de segurança
- Buscar a qualidade no atendimento do cliente

Em nosso trabalho, os objetivos da gestão estarão ligados a questões quantitativas de quanto pedir e quando pedir, para, assim, buscar a qualidade no atendimento do cliente. Os

parâmetros referentes a busca por melhores preços ou atendimento aos níveis de segurança não serão, por ora, considerados.

1.1.2.1 Quanto Pedir?

A quantidade de material requisitada está relacionada ao porte da operação, a técnica de gestão e a demanda. Essa decisão deve levar em consideração o equilíbrio entre o custo dispendido na compra, ou no processo de requisição de fornecimento, e o custo de estoque.

Segundo [22], podemos destacar alguns itens importantes quando se trata de custo de estoque e compra:

- Custo de colocação de pedidos
- Custo de desconto de preço
- Custo de falta de estoque
- Custo de Capital de Giro
- Custo de armazenagem
- Custo de obsolescência
- Custo de ineficiência de produção

A gestão de estoque deverá se basear no planejamento estratégico da corporação para tomar decisões, a fim de equilibrar os custos acima citados.

Como estratégia empresarial e filosofia corporativa, pode-se citar o *Just in Time* (JIT) como uma abordagem que visa reduzir ao limite mínimo o nível de estoque. Essa metodologia de trabalho, desenvolvida no Japão, pode ser melhor estudada em [22, 35]

1.1.2.2 Quando Pedir?

Essa decisão, segundo [35], envolve a análise da confiabilidade e do tempo para entrega do fornecedor. Além disso, deve levar em consideração a estratégia de operação do gestor, podendo ele decidir entre pedir continuamente ou em intervalos periódicos, como visto em [22]

O tempo de entrega (*lead time*) é fundamental para determinar quando pedir. Ele é determinado como o intervalo de tempo entre a efetivação do pedido e a validação da entrega do produto.

Para essa decisão também é interessante que se possua um modelo de previsão de demanda, permitindo o casamento entre a taxa de requisição de material e a taxa de consumo.

1.1.2.3 Sistema de gerenciamento

O aumento da complexidade da operação exige a utilização de técnicas mais eficientes, porém custosas, de gestão do estoque. Sistemas de informação serão necessários para garantir o registro das transações, para a geração de pedidos, validação da entrega correta e atendimento aos padrões de qualidade contratados.

O presente trabalho pode ser entendido como um possível "pacote" anexo ao sistema de gerenciamento. Seria possível, com os controladores aqui estudados, o registro automático de pedidos, necessitando (ou não) apenas da autorização do gerente (depois de uma verificação dos resultados apresentados pelo controlador).

1.2 O Modelo de Estoque

Neste trabalho, analisaremos, por meio de modelos matemáticos e simulações, um modelo de estoque baseado naquele apresentado e estudado em [12]. O sistema em questão pode ser entendido como um subsistema de uma cadeia de suprimentos que inclui o fornecedor de matéria-prima, a indústria fabricante do produto (junto a seus processos de operação), o estabelecimento intermediário (como centro de distribuição) e o estoque final, como em uma loja (Figura 1.1). Alguns trabalhos utilizam técnicas de controle para projetos que englobem toda a cadeia [25, 3]. Porém, esse trabalho foca a sua atenção aos procedimentos de um estágio da cadeia, como, por exemplo, a indústria fabricante.

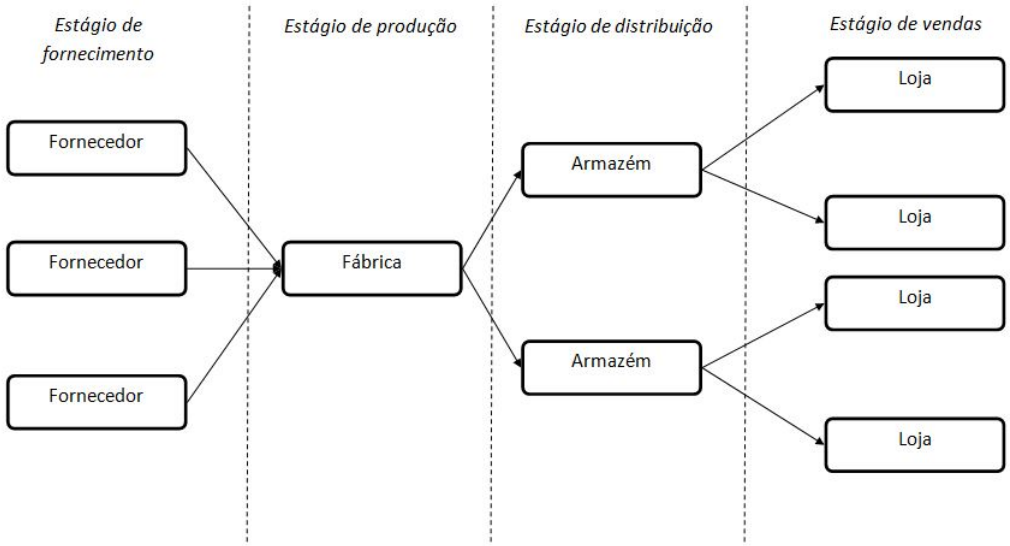

Figura 1.1: Cadeia de Suprimentos Multiestágio

Nela, o setor comercial requisita ao departamento de produção um determinado volume de produtos. A produção, por sua vez, necessita de um prazo para entregar os itens requeridos. De posse dos produtos recém chegados, o setor comercial pode despachar os produtos para o próximo nó da cadeia. A soma de todos os produtos em produção é denominada *Work-in-Progress*

Uma análise semelhante poderia ser feita para uma loja que possua a possibilidade de fazer pedidos diariamente (Figura 1.2). Sempre que julgasse necessário, o gerente da loja requisitaria ao seu fornecedor um determinado volume de um produto. Assim, o fornecedor teria um determinado prazo para a entrega do pedido. A soma de todos os pedidos, inclusive os que já foram despachados pelo fornecedor mas ainda não chegaram ao estoque da loja, é denominado *shipped*.

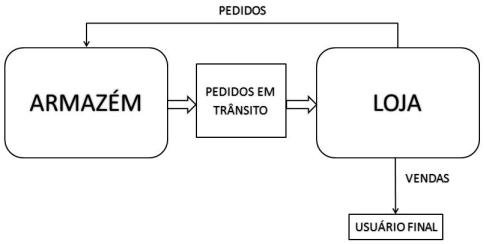

Figura 1.2: Esquema Armazém-Loja

Interessante notar que a estrutura dos dois exemplos é semelhante. Inclusive, *Work-in-*

Progress e *shipped* possuem mesmas características. Ambas as situações podem ser ilustrados pelo diagrama de fluxo da Figura 1.3

Desse modo, o sistema, de tempo discreto, será constituído pelo número de itens em estoque ($store$ ou $I(k)$), pela carga de venda diária ($selling$ ou $S(k)$), pelo processo de recebimento de produtos no estoque ($receiving$ ou $O(k-\tau) = R(k)$), pela demanda ($D(k)$), pelo número de itens pedidos em um determinado dia ($ordering$ ou $O(k)$) e pelo total de itens em transferência em determinado dia ($shipped$ ou $W(k)$). Após a realização dos pedidos, ocorre um atraso na entrega destes a central de estoque, o prazo de entrega (τ).

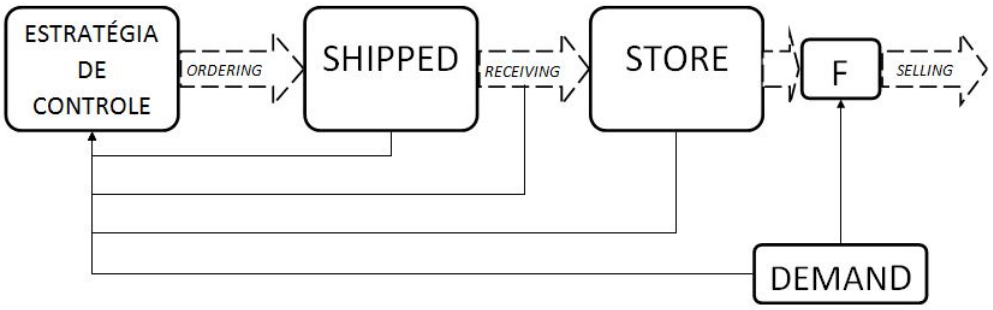

Figura 1.3: Arquitetura de estoque

Matematicamente, teremos, para $k > \tau$:

$$I(k) = I(k-1) + O(k-\tau) - S(k) \tag{1.1}$$

$$W(k) = W(k-1) + O(k) - O(k-\tau) \tag{1.2}$$

$$receiving(k) = ordering(k-\tau)$$

$$\text{por isso: } R(k) = O(k-\tau) \tag{1.3}$$

As variáveis $I(k)$ e $W(k)$ são saídas de dois sistemas integradores, sendo escolhas naturais para as variáveis de estado. A variável independente k representa o tempo discreto contado em dias. A quantidade de itens que chega ao estoque ($O(k-\tau)$) é a mesma quantidade pedida ($O(k)$) τ dias atrás. Dessa forma, o *lead-time* de nosso sistema será τ. Para [12], esse atraso é constante.

Importante ressaltar que nenhuma central de estoque possui espaço infinito. Assim, a variável $I(k)$ não poderá ultrapassar um limite preestabelecido. O que for pedido e acarrete em um valor maior que esse limite deverá ser descartado (ainda assim, representado custos ao gerente de estoque). Atingir o limite superior e ter que descartar produtos pode ser considerado uma situação ruim ao desempenho da empresa.

Sobre a dinâmica da variável $W(k)$, é interessante notar que, por representar o volume de todos os pedidos em trânsito (ou produção) até que cheguem ao estoque, ela pode ser matematicamente expressa por

$$\begin{aligned} W(k) &= O(k) + O(k-1) + \cdots + O(k-(\tau-1)) \qquad (1.4) \\ &= O(k) + \underbrace{[O(k-1) + \cdots + O(k-(\tau-1)) + O(k-\tau)]}_{W(k-1)} \\ &\quad -O(k-\tau) \\ &= W(k-1) + O(k) - O(k-\tau) \qquad (1.5) \end{aligned}$$

A equação 1.5 descreve a representação no espaço de estados para $W(k)$, cuja condição inicial é representada substituindo $k = 0$ na equação 1.4. Assim

$$W(0) = O(0) + O(-1) + \cdots + O(-\tau+1), \qquad (1.6)$$

Uma escolha razoável seria adotar

$$O(-1) = O(-2) = \cdots = O(-\tau+1) = 0 \text{ (zero)}, \qquad (1.7)$$

De tal forma que $W(0) = O(0)$.

Para melhor compreensão, o sistema funciona do seguinte modo: inicia-se pelo cálculo do sinal de controle, a quantidade pedida ($O(k)$). Entre a emissão do pedido e a entrega do produto ao estoque, ocorre um atraso, suposto fixo nesse trabalho, denominado *lead-time*. A soma de todos os pedidos em trânsito é denominada *Work-in-Progress*, $W(k)$. Passado o prazo de transferência, o item pedido passa a ser denominado $receiving$ e pode, agora, ser entregue ao estoque ($I(k)$). Este atualiza-se levando em consideração a quantidade recebida e a quantidade vendida ($S(k)$).

Agora, analisaremos a quantidade de vendas ($S(k)$). Ainda em [12], a quantidade vendida deverá ser igual a demanda pelo produto no dia k, caso a diferença entre estoque e demanda ($D(k)$) seja superior a 1, para que seja sempre possível ter, ao menos, um item como amostra para consumidores futuros.

Assim, matematicamente, teremos:

$$S(k) = f(D(k)) = \begin{cases} D(k) & \text{se } I(k-1) - D(k) > 1 \\ I(k-1) - 1 & \text{caso contrário} \end{cases} \tag{1.8}$$

Essa equação representa a parcela não linear da dinâmica de estoque. Em [12],a demanda é modelada como uma variável aleatória caracterizada por uma distribuição de *Poisson*. Além disso, é importante ressaltar que $S(k)$ é, também, uma função da variável de estado $I(k)$.

Assim, em resumo, temos um sistema discreto modelado por um duplo integrador, contendo atraso de transporte, não linearidade e entrada aleatória, constituindo, assim, um desafio de controle a ser abordado nesse trabalho.

Nota 1.1 *O lead-time na prática não é constante. Porém, nesse trabalho será considerado de tal forma, assim como em [12], por se tratar de um atributo que poderia aumentar consideravelmente a complexidade do estudo.*

1.2.1 *Economic Value Added* - EVA

Para que seja projetada uma estratégia de controle ótimo, uma função custo deve ser considerada. Assim como em [12], no caso deste trabalho, o *Economic Value Added* (EVA) será a fonte de comparação entre as estratégias elaboradas.

O EVA é, segundo [12], um medidor de desempenho financeiro, desenvolvido por Sloan e Stern. Ele é capaz de mensurar quanto de valor uma empresa é capaz de gerar aos seus investidores em um determinado período. Ele é composto de dois termos, Lucro Líquido e o Investimento Dispendido. Dessa forma, aumentar o EVA significa aumentar as receitas e/ou reduzir os gastos, que é uma estratégia clássica.

Assim, temos:

$$DailyEVA(k) = NetOperatingProfitAfterTax(k) - CapitalCharge(k) \tag{1.9}$$

O Lucro líquido operacional após o imposto (*Net Operating Profit After Tax* - NOPAT) é tido como:

$$NOPAT(k) = (SalesProfit(k) - OperatingExpenses(k))(1 - TaxRate) \tag{1.10}$$

O lucro decorrente das vendas ($SalesProfit$) é descrito por:

$$SalesProfit(k) = Sales(k) \cdot SalesPrice \tag{1.11}$$

Já os custos operacionais são resultantes da manutenção do estoque e dos capital dispendido no processo de pedido (ou compra) dos produtos. Assim,

$$OperatingExpenses(k) = HandlingCost(k) + BuyCost(k) \tag{1.12}$$

O custo de manutenção $HandlingCost$ é expresso por

$$HandlingCost(k) = UnitHC \cdot O(k - \tau) \tag{1.13}$$

onde $UnitHC$ é a constante que quantifica o valor dispendido na manutenção de cada item que chega ao galpão de armazenagem.

De forma análoga, $BuyCost$ é descrito por

$$BuyCost(k) = UnitCost \cdot O(k - \tau) \tag{1.14}$$

onde $UnitCost$ é o custo da realização do pedido de um produto.

O Capital Investido deverá ser expresso como a perda de valor aos acionistas, devido ao fato da organização ter recursos empatados que poderiam estar sendo utilizados para trazer riquezas aos investidores. É, assim, decorrente do custo de oportunidade.

$$CapitalCharge(k) = CostOfCapital \cdot TiedUpInCapital \tag{1.15}$$

Existem duas formas de uma companhia financiar seus investimentos, por empréstimos bancários ou utilizando seu próprio lucro (reduzindo o montante transferido aos acionistas).

Em nosso modelo, o capital investido é descrito por

$$CapitalCharge(k) = (InventoryCost(k)+StorageCost(k))ExpectedReturn+OoSCost(k) \tag{1.16}$$

O $InventoryCost$ é a parcela de capital que representa os custos de armazenagem, matematicamente expresso por

$$InventoryCost(k) = I(k) \cdot (UnitCost + HandlingCost(k)) \tag{1.17}$$

É interessante perceber que a quantidade de itens armazenada tem uma parcela de contribuição grande no cálculo do EVA. Ter produtos em excesso significa altos custos, por consequência redução nos lucros.

O $StorageCost$ é definido como os custos relacionados aos picos de estoque, logo relaciona-se a atingir um nível de estoque superior ao anterior. Assim,

$$StorageCost(k) = UnitStoreCost \cdot PeakStore(k) \tag{1.18}$$

onde $UnitStoreCost$ descreve a constante que quantifica o capital dispendido em cada item que ultrapassa o nível de estoque anterior. A variável $PeakStore$ é definida como

$$PeakStore(k) = PeakStore(k-1) + Peaking(k) \tag{1.19}$$

onde $Peaking$ é definido por

$$Peaking(k) = p(I(k)) = \begin{cases} I(k) - PeakStore(k-1) & \text{se } I(k) > PeakStore(k-1) \\ 0 & \text{caso contrário} \end{cases} \tag{1.20}$$

Voltando a equação 1.16, a taxa de retorno $ExpectedReturn$ descreve a porcentagem do capital investido que se espera que retorne aos investidores.

Ainda é necessário explicar uma importante parcela da equação 1.16. Como foi dito no início desse capítulo, não ser capaz de atender os consumidores significa perda de valor a empresa. Assim, o termo $OoSCost$ descreve o custo de perder a chance de concretizar vendas devido a falta de produtos disponíveis em estoque. Assim,

$$OoSCost(k) = t(I(k), S(k)) \begin{cases} TurnAwayCost \cdot TurningAway(k) & \text{se } I(k) = 1 \text{ e } S(k) > 0) \\ 0 & \text{caso contrário} \end{cases} \tag{1.21}$$

A equação 1.21 é interpretada da seguinte forma: caso o estoque esteja em seu limite mínimo (um) e haja um volume de vendas, isso significa que existia demanda[1] e, possivelmente, estamos deixando de vender novos produtos. Dessa forma, existe uma constante, $TurnAwayCost$ que quantifica o valor perdido ao deixar de vender um produto por este estar em falta no estoque. Já o termo $TurningAway$ é descrito pela diferença entre a demanda e o volume de vendas. Na prática é difícil de se quantificado, mas, mesmo assim, significa o custo de oportunidade em não se ter atendido todos os consumidores a procura do produto. Assim,

$$TurningAway(k) = D(k) - S(k) \tag{1.22}$$

É necessário calcular o real valor do EVA, que é uma soma dos valores de EVA diários. Assim

$$EVA(k) = EVA(k-1) + DailyEVA(k) \tag{1.23}$$

Temos, por fim, o EVA em função das variáveis da planta

$$\begin{aligned} EVA(k) &= EVA(k-1) \\ &+ (S(k) \cdot SalesPrice - (UnitHC \cdot O(k-\tau) + UnitCost \cdot O(k-\tau))(1 - TaxRate)) \\ &- \{[I(k)(UnitCost + O(k-\tau)UnitHC) \\ &+ UnitStoreCost \sum_{j=1}^{k} p(I(k))]ExpectedReturn + t(I(k), S(k))\} \end{aligned} \tag{1.24}$$

Nota 1.2 *Foi inicialmente testado um modelo mais simples para o EVA, no qual apenas o nível de estoque influenciava nos custos operacionais. Também, o capital investido era constante.*

Para tornar mais sucinto o capítulo de resultados, apenas ilustraremos os desempenhos das estratégias referentes a função objetivo descrita na seção 1.2.1.

[1] Só podemos concretizar a existência da demanda a partir da quantidade de vendas

1.2.2 Observações Gerais e Revisão da Literatura

Muitos trabalhos têm sido redigidos sobre o tema controle de estoque. Alguns com um foco semelhante ao adotado nesse trabalho, estudando um estágio da cadeia de suprimentos, outros, com maior abrangência, analisam a dinâmica da cadeia como um todo.

Uma análise minuciosa da evolução do tema na academia é feita em [23]. A leitura atenta mostra que existem trabalhos realizados na década de 50. Fazem 60 anos, mas é notável o "aquecimento" recente do assunto.

Pesquisadores têm desenvolvido estudos acerca de um controlador tipo APIOBPCS (*Automatic Pipeline Inventory and Order Based Production Control Systems*). O controlador APIOBPCS básico, como visto em [34, 11], pode ser descrito pela Figura 1.4. Em [34] é feito um estudo de arquiteturas para a estratégias de controle, de forma a garantir um desempenho satisfatório. Como proposta, adiciona-se, ao APIOBPCS usual, um controlador PID (Proporcional, Integral e Derivativo) e a previsão de demanda é feita com Filtros de Kalman Estendido.

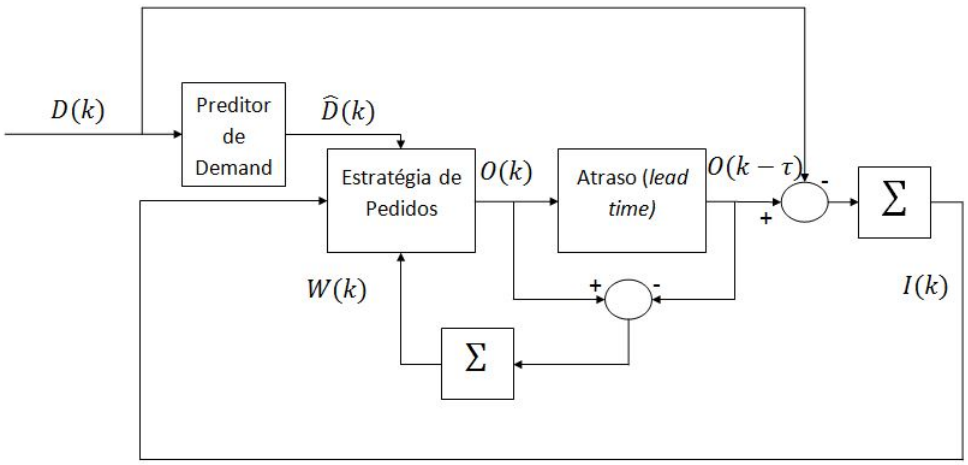

Figura 1.4: Controlador APIOBPCS básico - [34]

O estudo da dinâmica de estoque controlada por um PID também é realizado em [27]. Nele, inclusive, o sistema projetado é utilizado como plataforma didática a alunos recém ingressos na universidade.

Em [11], um caminho diferente é adotado. Estuda-se a dinâmica do sistema de estoque em resposta a uma tomada de decisão estilo gatilho (*trigger*). A estratégia de controle consiste em

medir a sua entrada e, quando esta atinge um determinado nível, ocorre o disparo de uma saída. Por exemplo, quando o nível de estoque for reduzido a um determinado patamar, o controlador gera um sinal de pedidos. Essa é uma técnica comumente utilizada pelos gerentes de estoque.

1.2.2.1 Medidores de Desempenhos

Nesse trabalho, o EVA será adotado como índice de desempenho. A capacidade de rastreamento de referências não é um fator inicialmente importante.

Existem outros índices de cunho financeiro, como o Retorno de Investimento (ROI - *Return of Investiment*), no qual o prazo para que se acumule, em receitas, o mesmo montante investido deve ser minimizado.

Já em [34], é feita uma função de medição de desempenho que envolve distribuição de pesos para a quantidade em estoque, a quantidade de vendas perdidas no ano e o erro da média do estoque para a referência desejada.

Em [6], também são citados a capacidade de manter o estoque em níveis desejados, a capacidade de atenuação no ruído da demanda e a robustez a variação do *lead-time*.

1.2.2.1.1 Medição das vendas perdidas Medir o volume de vendas não concretizados pode ser importante na análise da estratégia adotada. Afinal, clientes insatisfeitos trazem prejuízos em curto, médio e longo prazo. Assim, deve-se estimar o custo de oportunidade dispendido na perda de uma venda, normalmente por falta de produtos em estoque.

Nesse trabalho, a quantidade de vendas não concretizadas é um termo da função objetivo. Devemos optar por um controlador que atinja seus objetivos primários (maximizando o EVA) e ao mesmo tempo impeça a perda de valor da marca da empresa perante aos consumidores.

1.2.2.1.2 Efeito *bullwhip* Também é importante analisar o sinal de controle, o $ordering(k) = O(k)$, no que tange a sua amplitude máxima, seu espectro de frequência e sua variância.

A medição da amplificação da variância no sinal de controle é denominado *bullwhip* . Muitos trabalhos focam seus esforços em atenuar essa variação frequencial no $ordering$.

Para [4], *bullwhip* é decorrente da utilização do sinal de demanda estimada na estratégia de controle, da consideração de *lead time* não nulo, em se fazer pedidos em batelada e na

flutuação dos preços dos produtos.

Em [5] é feito um estudo das diferentes possibilidades de se medir *bullwhip* . Inicia-se com análises frequenciais e, posteriormente, estatísticas. A utilização de modelo estatístico, com a variância, também ocorre em [37].

Nesse trabalho, por simplicidade, o termo *bullwhip* não foi incluído na função custo EVA, mas é importante analisar o sinal de controle como parâmetro de estudo da eficiência do controlador.

1.2.2.2 Adoção de Algoritmos Genéticos

Para o modelo de estoque, podemos adotar o seguinte vetor de estados:

$$\mathbf{x}(k) = \begin{bmatrix} I(k) \\ W(k) \end{bmatrix} \tag{1.25}$$

Podemos, então, escrever o modelo no espaço de estados da seguinte forma:

$$\begin{aligned} \mathbf{x}(k) &= \mathbf{x}(k-1) + \begin{bmatrix} O(k-\tau) \\ O(k) - O(k-\tau) \end{bmatrix} + \begin{bmatrix} -f(D(k)) \\ 0 \end{bmatrix} && (1.26) \\ &= G(\mathbf{x}(k-1), O(k), \tau) + L(k) && (1.27) \end{aligned}$$

Manipulando a equação 1.27, podemos chegar a seguinte equação matricial

$$\mathbf{x}(k) = \mathbf{x}(k-1) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) + \begin{bmatrix} 1 \\ -1 \end{bmatrix} u(k-\tau) + \begin{bmatrix} -1 \\ 0 \end{bmatrix} L(k) \tag{1.28}$$

onde $u(k) = O(k)$.

Note que a entrada exógena $D(k)$ passa pela função $f(\cdot)$ não linear, definida em 1.8. O estado é $\mathbf{x}(k)$, o sinal de controle $O(k)$ e o distúrbio é expresso por $L(k)$ (função da variável aleatória, $D(k)$).

Dado um horizonte final, denotado $k_{horizonte}$, o problema de controle ótimo é:

$$\begin{aligned} &\max_{O} \; EVA(k_{horizonte}) \\ \text{s.a } &\mathbf{x}(k) = G(\mathbf{x}(k-1), O(k), \tau) + L(k) \end{aligned} \tag{1.29}$$

Por outro lado, podemos especificar uma estrutura de controle (por exemplo, PID) e parametrizar este controle por seus ganhos e referências, colocados em um vetor denotado ζ. Neste caso, podemos escrever:

$$O(k) = \varphi(\zeta, \mathbf{x}(k), \hat{D}(k)). \tag{1.30}$$

Com esta escolha de controlador, podemos formular o seguinte problema de otimização:

$$\begin{aligned} &\max_{\zeta} \; EVA(k_{horizonte}) \\ \text{s.a } &\mathbf{x}(k) = G(\mathbf{x}(k-1), \varphi(\zeta, \mathbf{x}(k), \hat{D}(k)), \tau) + L(k) \end{aligned} \tag{1.31}$$

É importante ressaltar que 1.31 é um problema de horizonte finito, no qual somente o valor final da função objetivo é maximizado e este é o foco do trabalho.

Note que é necessário solucionar o problema 1.31 para calcular ζ, e assim, definir completamente a lei de controle.

Dada a relativa complexidade da função objetivo (escolhido como EVA), optou-se pela utilização de um algoritmo genético para a solução do problema de controle ótimo.

Nota-se, também, que a utilização de algoritmos genéticos é recorrente em artigos de pesquisa na área. Em [25] utiliza-se algoritmos genéticos para resolver o problema de estoque na cadeia de suprimentos, mas em um contexto diferente. Ao invés de otimizar a resposta de um único sistema inserido na cadeia de suprimentos (um estágio), ele procura otimizar os custos de estoque em toda a cadeia, e não utiliza a abordagem de sistema dinâmico. Em [3] utiliza-se um procedimento similar, apesar de considerar também custos por falta de produtos.

1.3 Objetivo

O objetivo desse trabalho é desenvolver uma lei de controle que otimize o EVA de um sistema de gestão de estoque ao final de um período especificado. Para tanto, serão estudadas estratégias diversas que serão analisadas por meio de simulações realizadas em plataforma MATLAB.

1.4 Motivação

O desenvolvimento da Tecnologia de Informação e sua ampla utilização nos meios de gestão permite o projeto de estruturas mais complexas, criadas para auxílio aos tomadores de decisão.

Entende-se que a utilização de técnicas de controle em sistemas de estoque é um amplo campo para os acadêmicos e pesquisadores da grande área de Controle, assim como um potencial produto de novos negócios para a área de TI.

Em diversos trabalhos [34, 27, 23], as técnicas clássicas de controle foram examinadas e simuladas por seus autores.

Já, nesse trabalho, serão discutidos temas como Controle por Lógica Fuzzy, Algoritmos Genéticos, Sistemas de Predição por suavização exponencial, além dos clássicos PIDs.

1.5 Conteúdo do Trabalho

O trabalho é dividido em 5 capítulos. O primeiro, este, visa contextualizar o leitor no sistema estudado, na sua importância para um determinado negócio, suas complexidades e características. Esse capítulo também é responsável por garantir os limites desse trabalho, bem como a motivação dos autores em realizá-lo.

O segundo capítulo deverá ser utilizado como referência teórica dos assuntos abordados nesse trabalho. Temas fundamentais à tese como teoria de controle clássico e realimentação de estados, bem como controle por Lógica *Fuzzy* serão discutidos. Além disso, os conceitos de Controle Ótimo, Algoritmos Genéticos e Técnicas de Predição serão estudados.

O capítulo 3 trata da engenharia de solução do problema proposto. O passo-a-passo é descrito, bem como as dificuldades encontradas, plataformas de simulação utilizadas e os motivos para a adoção de MATLAB. Nele as estratégias de controle são desenvolvidas e explicadas.

As simulações, os resultados e as comparações são feitas no capítulo 4.

O capítulo 5 contém análise final dos resultados obtidos e por apontar o futuro de novos trabalhos na área.

O funcionamento da caixa de ferramenta, utilizada para simulações com o Algoritmo Genético, é apresentado no apêndice A.

Capítulo 2

Revisão Teórica

Este capítulo visa fornecer os requisitos teóricos necessários ao compreendimento das estratégias de solução propostas nesse trabalho. Os assuntos aqui tratados não serão minuciosamente discutidos, mas abordados de forma sucinta.

Nesse capítulo iniciaremos abordando alguns aspectos da Teoria de Controle relevantes ao sistema que estudamos. Serão abordados conceitos de Controle por Lógica *Fuzzy*, como alternativa a sintonia comum de controladores. Da mesma forma, Algoritmos Genéticos serão brevemente apresentados, como alternativa às ferramentas usuais de otimização.

Versa-se, também, sobre as metodologias de predição, importantes para a estimação da demanda futura pelos produtos do estoque.

2.1 Revisão de aspectos de Controle

2.1.1 Características de Controle Discreto

Em controle digital, o modelo discreto está em uma forma própria a ser implementada em um computador. Tanto para a o cálculo da estratégia de controle (como em um microcontrolador), quanto para simulações numéricas.

É comum também, gerar equações discretas a partir de transformações diversas feitas em modelos contínuos. É necessário particular cuidado aqui, principalmente no que diz respeito a estabilidade do sistema.

Diversos métodos da Teoria de Controle Linear Contínuo possui um espelho no tempo discreto. Lugar da raízes, análises frequencial, critérios de estabilidade de Jury (em paralelo ao

de Hurwitz, para modelos contínuos). É igualmente comum encontrarmos aplicações discretas decorrentes de transformações em modelos contínuos. Assim, o Engenheiro trabalha com ferramentas clássicas e transfere ao domínio discreto por meio de transformações como o método de Euler Atrasado

Definição 2.1 *Método de Euler Atrasado (Backward Euler)*

Como explica [8], na transformação Backward Euler (ou diferenciação atrasada), o termo derivativo é substituído da seguinte forma

$$\dot{x} = \frac{1}{h}[x(k) - x(k-1)] \tag{2.1}$$

onde h é a frequência de amostragem do sistema[1]*. Aplicando 2.1 novamente, a segunda derivada é representada por*

$$\ddot{x} = \frac{1}{h^2}[x(k) - 2x(k-1) + x(k-2)] \tag{2.2}$$

O controlador a três termos convencional pode ser transformado para uma versão discreta. Primeiramente, é necessário obter uma versão derivativa do PID Contínuo para eliminarmos o termo integral. Assim

$$\dot{u}(t) = K_p\dot{e}(t) + K_i e(t) + K_d\ddot{e}(t) \tag{2.3}$$

Aplicando 2.1 e 2.2 em 2.3, com frequência de amostragem $h = 1$, tem-se

$$\begin{aligned} u(k) - u(k-1) &= K_p[e(k) - e(k-1)] + K_i e(k) + K_d[e(k) - 2e(k-1) + e(k-2)] \\ u(k) &= u(k-1) \\ &+ K_p[e(k) - e(k-1)] \\ &+ K_i e(k) \\ &+ K_d[e(k) - 2e(k-1) + e(k-2)] \end{aligned} \tag{2.4}$$

[1] A frequência de amostragem possui grande importância para a teoria de controle digital. O leitor pode encontrar em diversos textos [8, 21, 16] referências sobre como escolher e trabalhar com a frequência de amostragem

2.2 Introdução ao Controle por Lógica *Fuzzy*

Um Engenheiro de Controle é um profissional bem capacitado matematicamente, e deve, em sua rotina, ser capaz de extrair análises equacionais complexas sobre os sistemas em que trabalha. Suas tarefas iniciam, comumente, com a modelagem matemática do sistema. Após, faz-se o uso de tal modelo para o projeto do controlador. Então, é possível realizar estudos sobre a malha fechada do sistema. Somente quando todos os requisitos de projeto estão garantidos, o controlador deve ser implementado e, se possível, testado em funcionamento no sistema.

E quando o modelo matemático é demasiadamente complexo? O início da cadeia do projeto do controlador está comprometido.

A Lógica *Fuzzy* é justamente a metodologia aplicável em tais casos. Por mais que um modelo em forma de equações seja importante, em muitos casos ele é difícil de ser extraído. Assim, a Lógica *Fuzzy* e sua vertente, o Controle *Fuzzy*, mantém o foco no primordial, o entendimento intuitivo.

Não só quando a modelagem é complicada que a Lógica *Fuzzy* é útil. No geral, sempre que houver incerteza. Em [29], ela é vista como uma poderosa ferramenta para utilização em ambientes incertos e/ou ambíguos.

A Lógica *Fuzzy* tem, então, amplo campo de aplicabilidade quando existem informações indeterminadas. Conquista sucesso em processos em que o raciocínio humano e suas vertentes estão envolvidas. Não é exagero dizer que sua raiz está no conhecimento humano e na linguagem natural. Sem modelagem matemática complexa, apenas a experiência técnica sendo analisada e utilizada por essa importante metodologia de engenharia.

Sem dúvida, *Fuzzy* não é o remédio para todos os males. Em [29], discute-se a incapacidade da sua utilização em projeto nos quais precisão é exigida. Além do mais, quando técnicas matemáticas não se mostram complexas demais ou apresentam resultados satisfatórios, não existe razão para aumentar a complexidade computacional de um sistema de controle utilizando técnicas *Fuzzy*, como mostra [24].

2.2.1 Projeto de um Controlador Fuzzy

Para [32], a utilização de um controlador *fuzzy* foi, pela primeira vez, utilizado por Mandami em 1976, ao controlar uma planta cementeira.

Um controlador, sintonizado po Lógica *Fuzzy*, será extremamente baseado na experiência profissional dos que trabalham com o sistema foco. Essa habilidade e conhecimento sobre o processo deverá ser transmitida para um conjunto de regras, por meio da linguagem natural. Os complexos modelos sedem lugar a termos como “muito frio” ou “pouco rápido”. A Figura 2.1 é uma forma representativa que tenta ilustrar as características desse tipo de controlador.

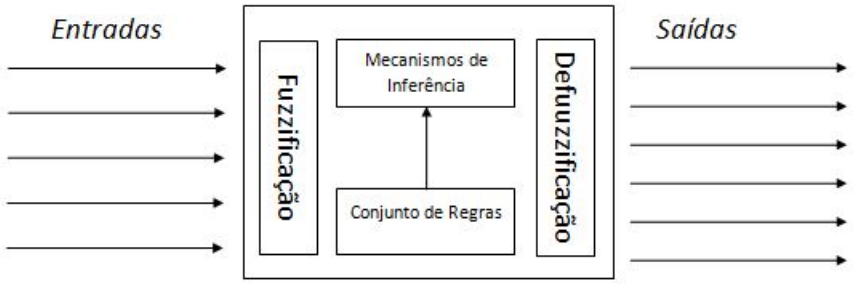

Figura 2.1: Esquema de um controlador *Fuzzy* [24]

A elaboração da estratégia de controle *Fuzzy* se inicia com a definição das entradas e saídas. Quando se deseja controlar a velocidade de um carro, como exemplificado em [36], a entrada por ser a própria velocidade. A saída pode ser a aceleração.

Já a estrutura interna do controlador é definida, em [24], pelas técnicas de *fuzzificação* e *defuzzificação*, pelo conjunto de regras e pelo mecanismo de inferência.

O conjunto de regras abrange o conhecimento sobre o sistema de uma forma simples e direta, sob a forma SE-ENTÃO. O mecanismo de inferência é a interpretação dos dados de entrada e do conjunto de regras para a tomada de decisão.

Agora, para que o controlador funcione é fundamental que todos seus elementos estejam em um mesmo domínio, o *Fuzzy*. Para isso, é necessário interpretar a entrada e transformá-la, por técnicas apropriadas discutidas na próxima seção, em *Fuzzy*. O contrário ocorre na saída do controlador, quando este deve enviar sinais físicos a planta. Estes elementos são os denominados *fuzzificação* e *defuzzificação*, respectivamente.

Voltando ao exemplo do controle da velocidade do carro, podemos entender a própria

velocidade como entrada que deve ser mapeada ao domínio nebuloso pela *fuzzificação*. Assim, o valor resultante desse cálculo é analisado pelo mecanismo de inferência para saber quais regras do conjunto se adequam. Por exemplo, caso a velocidade esteja demasiadamente baixa, por ser útil a seguinte regra:

SE *velocidade* = *baixa* ENTÃO *aceleração* = *alta*

A aceleração, por sua vez, deve ter seu valor físico calculado a partir do método escolhido para a *defuzzificação*.

2.2.2 Conceitos Fundamentais

A Lógica *Fuzzy* e sua álgebra têm uma enorme extensão de definições, lemas, corolários e muitas técnicas. Aqui, serão discutidos somente os conceitos fundamentais ao entendimento do controlador *fuzzy* utilizado nesse projeto. Para melhor entendimento do mundo nebuloso, os leitores são recomendados a pesquisarem referências mais específicas, como [29, 24, 36, 38]. Para os que necessitam ter uma melhor visão sobre algoritmos de controle *fuzzy*, a consulta a [15] é uma ótima opção.

Em [24], um sitema *Fuzzy* é definido como um mapeamento estático entre suas entradas e suas saídas. Tantos suas entradas, quanto suas saídas são números reais, ou melhor, pertencem ao domínio "crisp".

Entende-se por "crisp" os conjuntos clássicos com fronteiras bem definidas. Em "crisp", a cor de um objeto, por exemplo, é vermelha ou não vermelha. A cor laranja, portanto, seria não vermelha, assim como o azul ou o verde. O universo *Fuzzy* é mais abrangente. Seus conjuntos não possuem fronteiras tão claras e rígidas. A cor laranja, retornando ao exemplo, poderia pertencer, mesmo que parcialmente, ao conjunto vermelho.

2.2.2.1 Universo de Discurso

Em [29], o universo de discurso é entendido como o conjunto de toda a informação compreendida em um determinado problema. Formalmente, os conjuntos crisp $\mathcal{U}_i$ e $\mathcal{Y}_i$ são denominados *universos de discurso*, ou *conjuntos universais*, para u_i e y_i. Esse conjuntos representam a união de todos os elementos possíveis para o contexto em análise.

Segundo o exemplo do carro, o universo de discurso $\mathcal{U}_i$ engloba todos os valores de velocidade possíveis, desde a nula até a máxima atingida pelo modelo do veículo.

Diz-se, então, que determinada velocidade u_i faz parte do universo de discurso $\mathcal{U}_i$. Matematicamente, $u_i \in \mathcal{U}_i$.

2.2.2.2 Valores Linguísticos

Alem dos valores "*crisp*", ou clássicos, assumidos pela variável u_i dentro do universo $\mathcal{U}_i$, em [24], introduz-se novas variáveis à análise do controlador, as variáveis linguísticas $\tilde{u}_i$ e $\tilde{u}_i$. Assim, $\tilde{A}_i^j$ representa um possível valor para a variável linguística $\tilde{u}_i$, definida sobre o universo $\mathcal{U}_i$. Caso assumamos, segundo [24], que $\tilde{u}_i$ possa assumir diversos valores linguísticos, esse será um do seguinte conjunto

$$\tilde{A}_i^j = \{\tilde{A}_i : j = 1, 2, \ldots, N_i\} \tag{2.5}$$

Retornando ao exemplo do carro, a variável linguística $\tilde{u}_i$, a velocidade do carro, poderia assumir os valores $\tilde{A}_1^1 =$ *"alta"*, $\tilde{A}_1^2 =$ *"média"* e $\tilde{A}_1^3 =$ *"baixa"*.

O mesmo critério pode ser utilizado para a saída do sistema y_i. Esta pode ser, linguisticamente, representada como $\tilde{y}_i$ e assumir valores pertencentes ao seguinte conjunto:

$$\tilde{B}_i^j = \{\tilde{B}_i : j = 1, 2, \cdots, M_i\} \tag{2.6}$$

2.2.2.3 Regras Linguísticas

As regras linguísticas são responsáveis por conectar causas e consequências. Melhor explicando, é o mapeamento *fuzzy* utilizado para transformar entradas em saídas. Funciona de acordo com a clássica metodologia *modus ponens*:

SE *premissas* ENTÃO *ação*

Premissas são entendidas, por [32], como uma proposição lógica cuja veracidade pode ser determinada. Assim, verificado o fato das entradas do sistemas estarem dentro das possibilidades de medição do controlador, é possível gerar uma ação com base no conjunto de regras linguísticas.

Podemos, agora, extrapolar a definição para um sistema com múltiplas entradas e uma saída. Dessa forma, segundo [24], sejam as entradas $u_i \in \mathcal{U}_i$ e saídas $y_i \in \mathcal{Y}_i$, representadas linguisticamente, por $\tilde{u}_i$ e $\tilde{y}_i$, respectivamente. Sejam, também, os valores linguísticos $\tilde{A}_i^j$ e $\tilde{B}_i^j$ possíveis de serem assumidos por $\tilde{u}_i$ e $\tilde{y}_i$, respectivamente. É possível, portanto, definir a estrutura *modus ponens* da seguinte forma:

$$\text{SE } \tilde{u}_1 \text{ é } \tilde{A}_1^j \text{ E } \tilde{u}_2 \text{ é } \tilde{A}_2^k \text{ E } \cdots \text{ E } \tilde{u}_n \text{ é } \tilde{A}_n^t \text{ ENTÃO } \tilde{y}_q \text{ é } \tilde{B}_q^p \tag{2.7}$$

Interessante notar que a estrutura de um controlador MIMO (múltiplas entradas e múltiplas saídas - *Multiple Inputs, Multiple Outputs*) pode ser repartida em diversas combinações MISO (múltiplas entradas e saída única - *Multiple Inputs, Single Output*), como mostra [24]. Assim, cada sinal de controle (saída da unidade de controle) pode ser interpretada como um sistema independente.

2.2.2.4 Função de Pertinência

Para [24], a lógica *fuzzy* e seus conjuntos são meios heurísticos para quantificar variáveis linguísticas, valores linguísticos e o conjunto de regras que é especificado.

A principal forma para quantificação de valores *fuzzy* é denominada *função de pertinência*. Ou, como define [17], *função característica* ou, ainda, *função de discriminação*.

Já discutimos as diferenças entre conjuntos "crisp" e *fuzzy* no que tange suas fronteiras. O primeiro, possui limitações claras, enquanto, o segundo, possui vagueza e pertinência crescente em suas bordas.

Assim, para um conjunto "crisp", poderíamos definir a função de pertinência de uma variável x em um conjunto A, como sendo:

$$\mu(x) = \begin{cases} 1 & \text{se } x \in A \\ 0 & \text{caso contrário} \end{cases} \tag{2.8}$$

Já uma função de pertinência *fuzzy*, necessita descrever a certeza com que $u_i \in \mathcal{U}_i$, com variável linguística $\tilde{u}_i$, pode ser linguisticamente classificada como $\tilde{A}_i$. Formalmente definindo, como encontrado em [24], temos:

$$\mu(u_i) : \mathcal{U}_i \mapsto [0, 1] \tag{2.9}$$

Por exemplo, a função de pertinência para a velocidade baixa do carro poderia ser expressa, graficamente, de acordo com a 2.2.

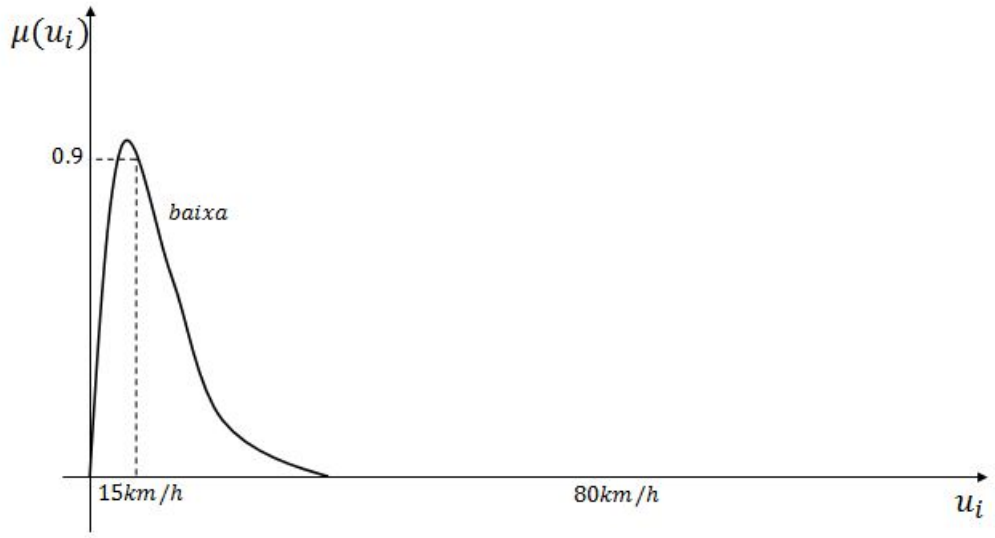

Figura 2.2: Função de Pertinência para o valor linguístico *baixa*

Caso o velocímetro marque 80 km/h, a pertinência da velocidade baixa seria nula. Diferentemente, caso o carro viaje a 15 km/h, a pertinência, ou seja quanto se aproxima da total veracidade do valor linguístico, atinge 0.9.

2.2.2.5 Conjuntos *Fuzzy*

Segundo [17], conjuntos *fuzzy* são mapeamentos de elementos "crisp", em [0,1], de acordo com funções de pertinência. Assim, sejam $\tilde{u}_i$ e $\tilde{A}_i$, variável e valor linguísticos de $u_i \in \mathcal{U}_i$, respectivamente. Assim como $\mu_{\underset{\sim}{A}_i}(u_i)$ a função de pertinência que transforma u_i. Um conjunto *fuzzy* $\underset{\sim}{A}_i^j$ pode ser definido, como expressa [24], como:

$$\underset{\sim}{A}_i^j = \{(u_i, \mu_{\underset{\sim}{A}_i^j} : u_i \in \mathcal{U}_i\} \tag{2.10}$$

Apesar da descrição em par ordenado ser amplamente utilizada, nesse texto, utilizaremos a seguinte forma:

$$\underset{\sim}{A}_i^j = \sum \mu(u_i)/u_i \tag{2.11}$$

Nessa equação, o somatório não representa a soma dos elementos, e sim a união dos elementos do conjunto.

2.2.2.6 *Fuzzificação*

Para [24], *fuzzificação* é o mecanismo que transforma $u_i \in \mathcal{U}_i$ em conjuntos *fuzzy*. Assim, como explica [32] *fuzzificar* significa achar uma versão *fuzzy* para um conceito "crisp". Ainda, é achar o grau de pertinência de uma variável linguística.

Portanto, seja $\mathcal{U}_i^*$ todos os possíveis conjuntos *fuzzy* que podem ser definidos em $\mathcal{U}_i$, a *fuzzificação* transforma $u_i \in \mathcal{U}_i$ em $\underset{\sim}{A}_i$ definido em $\mathcal{U}_i$. Logo:

$$\mathcal{F} : \mathcal{U}_i \mapsto \mathcal{U}_i^* \tag{2.12}$$

2.2.2.7 Mecanismos de Inferência

Em [24], mecanismos de inferência são explicados como responsáveis por duas tarefas em sistema *fuzzy*. A primeira, é determinar a extensão da relevância de cada regra dada a situação corrente, caracterizada pelas entradas u_i. A segunda, é gerar conclusões baseadas nas entradas u_i e no conjunto de regras.

Inicialmente, então, é necessário combinar o resultado do processo anterior, *fuzzificação* das entradas, com os conjuntos *fuzzy* das premissas do conjunto de regras. Assim, em sequência, é possível determinar o grau de pertinência de determinada premissa, medindo o grau de sua certeza.

Isso feito, em [29] três métodos são definidos para a geração de conclusões: Mamdani, Sugeno e Tsukamoto. Nesse trabalho, elucidaremos o método de Mamdani, que será utilizado no projeto de nosso controlador. Assim, [29], define o processo de conclusão por Mamdani da seguinte forma:

$$\mu_{\underset{\sim}{B}_k}(y) = \max_k[\min[\mu_{\underset{\sim}{A}_1^k}(u_1), \cdots, \mu_{\underset{\sim}{A}_n^k}(u_n)]] \tag{2.13}$$

2.2.2.8 *Defuzzificação*

Logo terminado o processo de inferência e, portanto, tendo posse da conclusão *fuzzy* decorrente do mesmo, devemos transformar esse resultado em um sinal de controle que a planta compreenda. Então, faz-se necessário mapear a saída nebulosa em um valor essencialmente "crisp".

A literatura, [24, 29, 36], discute diversos métodos utilizados para a transformação da conclusão *fuzzy* em um valor "crisp". E, também, não é difícil inventar novos métodos. Para a simplificação dessa seção, será explicado, aqui, o método de centro de gravidade, pois será utilizado no nosso projeto.

Assim, em [29], o método do centro de gravidade é definido como escolha de uma valor y_i "crisp" se utilizando do centro da área implicada pelo conjunto *fuzzy* inferido para a saída. Logo, matematicamente, temos:

$$y_i = \frac{\int \mu_{\underset{\sim}{B}}(y) \cdot y dy}{\int \mu_{\underset{\sim}{B}}(y) dy} \tag{2.14}$$

Nota 2.1 *Detalhes da álgebra fuzzy não são tratados nesse texto. Para melhor entender os operações com conjuntos nebulosos, os leitores são recomendados a consultar as referências citadas no início da sessão.*

2.2.3 Razões para utilização da Lógica *Fuzzy* nesse projeto

Para [15], algoritmos *fuzzy* para controle são fáceis de aprender e de serem utilizados. Isso, pois se aproximam da intuição humana.

Assim, a metodologia nebulosa se tornou interessante para o controle de um sistema não linear com grande influência da aleatoriedade. Particularmente, a modelagem do sistema de estoque não é uma tarefa árdua e é possível ter sucesso com incertezas pequenas. Porém, a imprecisão na determinação da frequência de vendas ($S(k)$) torna o controle *fuzzy* uma ferramente importante nesse projeto.

2.3 Introdução aos Algoritmos Genéticos

Algoritmos Genéticos foram propostos nos anos 70 por Holland [13] e são uma forma simples e efetiva de resolver problemas de otimização. São baseados em Indivíduos Artificiais [25] e na seleção natural de Darwin.

Esses algoritmos, da classe meta-heurísticos, apresentam, como vantagens ao nosso desafio, o fato de sua complexidade ser independente do problema a ser otimizado e que ele funciona em situações com variáveis contínuas ou discretas. Dessa forma, problemas que poderiam ser muito complexos ou até inviáveis computacionalmente para algoritmos tradicionais, se tornam mais tratáveis pelos algoritmos genéticos. A desvantagem desse tipo de algoritmo é que não há a garantia de um resultado ótimo.

Esse tipo de algoritmo possui alguns mecanismos, que de forma resumida, funcionam da seguinte maneira:

Uma população inicial de possíveis soluções deve ser gerada utilizando algum critério para tal. Todos os indivíduos dessa população devem ser codificados em um dado Alfabeto, cuja representação contenha o espaço de busca [26]. A cada subparte desse indivíduo codificado é dado o nome de gene. Cada gene possui um valor dentro do alfabeto de codificação. Cada valor possível desse alfabeto é denominado um alelo possível para o gene. A seguir, ocorre a reprodução dos indivíduos artificiais. Essa reprodução visa combinar as características de dois indivíduos para que dois novos sejam gerados. Para que isso ocorra, é necessário que exista um critério para selecionar quais serão os dois indíviduos geradores. Em geral essa seleção é feita sobre os resultados escalonados de suas performances medidas pela função objetivo, além de uma recombinação entre os genes de ambos. Após gerados um número de soluções igual ao tamanho da população, ocorre mutação de alguns genes, ou seja, seus alelos trocam de valor. Por fim, desse novo Universo de soluções possíveis, entre geração anterior e nova geração, devem ser escolhidos quais indivíduos sobreviverão e formarão a nova população. Para tanto, é necessário determinar se haverá elitismo, isso é, e as melhores soluções irão continuar na população futura. Por fim, por se tratar de um algoritmo que se repete iterativamente, precisamos de um critério de parada. No diagrama da Figura 2.3, podemos ter uma visão esquemática do funcionamento.

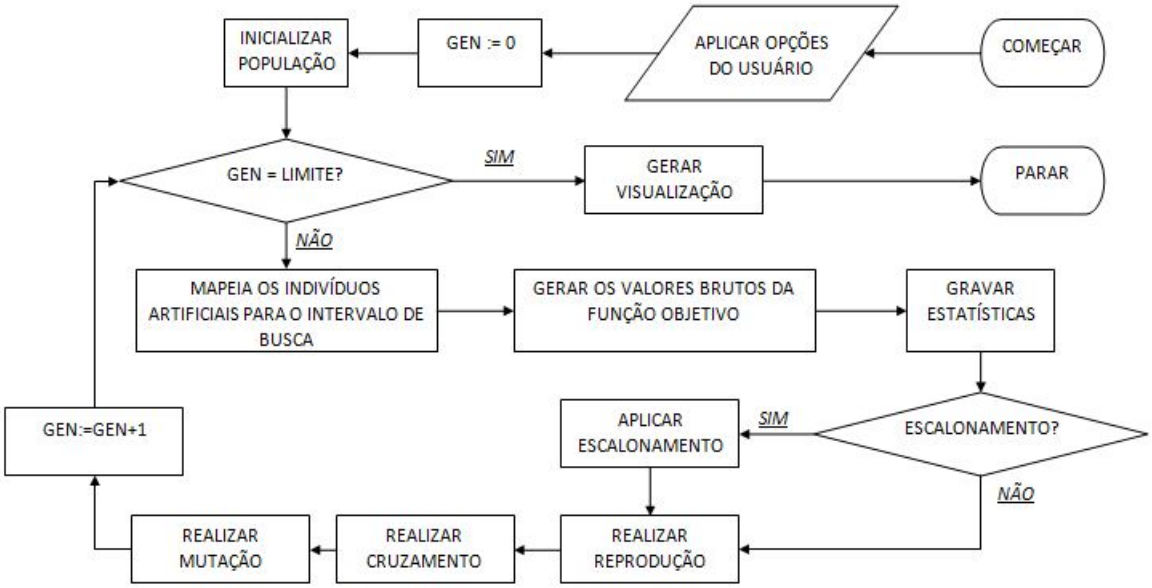

Figura 2.3: Diagrama esquemático do funcionamento dos Algoritmos Genéticos. [30]

Outra forma de fazer otimização utilizando Algoritmos Genéticos é utilizando subpopulações. Otimiza-se cada uma dessas subpopulações separadamente e, ao fim de n iterações, realiza-se migração de alguns indivíduos entre as populações. Essa migração também deve obedecer alguns critérios pré estabelecidos.

2.3.1 Geração da População Inicial

A população inicial, aquela que iniciará o algoritmo, é importante por diversas razões: É a partir dela que serão geradas as populações subsequentes, portanto a ausência de um alelo em algum gene se propagará para todas as outras populações a não ser que ocorra mutação em algum dos indivíduos naquele gene. Além disso, uma população que está toda concentrada em uma pequena porção do Espaço de Busca provavelmente fará o algoritmo ficar preso em um ótimo naquela região.

Portanto, a forma como ela será gerada é de fundamental relevância para o restante do algoritmo. Existem diversas formas de gerar essa população. A primeira, e mais simples, é gerar de forma aleatória. A vantagem desse método reside em sua simplicidade, ao passo que ele tem como desvantagem não garantir a cobertura total do espaço de busca e ainda ficar presa a algum vício do gerador de números pseudo-aleatórios do computador. Pode-se, também,

fazer um número qualquer de inicializações aleatórias e utilizar aquela que obteve a melhor performance [2]. Uma outra forma de criar a população inicial é por meio de uma grade. Vários subespaços são gerados dentro do espaço de busca e dentro de cada um desses subespaços é posicionado um determinado número m de indivíduos, onde $m < n$. Por fim, é possível utilizar um outro método heurístico a fim de encontrar bons pontos para serem alimentados aos Algoritmos Genéticos.

Além da forma, outro fator importante é o tamanho da população inicial. Para determina-lo, alguns trabalhos foram feitos [9] [10] [31]. Ao escolher o tamanho da população, nos encontramos em uma decisão de *trade-off* entre efetividade e eficiência[26]. Uma população muito pequena fará o algoritmo ser eficiente, no sentido de ser menos custoso computacionalmente e mais rápido. Porém, será pouco efetivo, pois a solução encontrada tenderá a não ser boa, ao passo que uma população maior provocará o contrário: bons pontos, possivelmente os melhores, serão encontrados, mas ao custo de um algoritmo muito moroso.

2.3.2 Codificação em um Alfabeto

A seguir, os indivíduos gerados precisam ser codificados em um vetor que contém símbolos de um dado alfabeto. Essa codificação é fundamental pois é ela que gera os genes, ou as características do indivíduo artificial. O processo é basicamente codificar os indivíduos em um vetor $\mathbf{b}$ de genes, onde esse vetor representa o indivíduo previamente codificado. Para o escopo desse trabalho, assim como para a maior parte das aplicações dos algoritmos genéticos, o alfabeto que estudaremos é o binário, ou seja, os alelos possíveis para cada gene serão os valores 0 e 1.

Ao codificarmos variáveis presentes em um espaço de busca precisamos estar atentos ao tamanho desse espaço e à precisão que desejamos para essas variáveis. Segundo [26], se temos um espaço de busca que vai de a até b e desejamos uma precisão minima de p, o número de bits l a ser utilizado será dado por

$$l = \log_2(\frac{b-a}{p} + 1) \tag{2.15}$$

2.3.3 Reprodução

2.3.3.1 Escalonamento

Para que o algoritmo de seleção escolhido funcione sempre a contento, é necessário escalonar os resultados brutos da função objetivo para cada indivíduo. Não fosse isso, seria muito mais difícil, para o algoritmo de seleção, funcionar de forma satisfatória quando os valores atingidos na função objetivo começarem a se tornar próximos, ou seja, no momento em que o Algoritmo Genético estiver se aproximando de convergir para um valor.

Nesse sentido, utiliza-se escalonamento para criar maior distinção entre valores próximos, seja em razão da convergência do algoritmo ou de um espaço de busca restrito. Assim como nas outras etapas do algoritmo, existem diversas formas diferentes de realizar o escalonamento. Para nosso trabalho, a forma utilizada será o Ranqueamento.

Nessa forma de seleção, o indivíduo mais adaptado, isso é, aquele que atingiu o melhor resultado na função objetivo no caso de maximização, ou o que atingiu o menor resultado no caso de minimização, recebe o ranqueamento de 1. O segundo mais apto recebe ranqueamento de 2 e assim sucessivamente. O valor escalado, então, deve obedecer a duas regras:

- O valor escalonado será proporcional a $\frac{1}{\sqrt{n}}$, onde n é o ranqueamento do indivíduo
- A soma dos valores escalonados será igual ao número de indivíduos que serão selecionados pelo algoritmo de seleção

2.3.3.2 Seleção

Existem diversos critérios de seleção daqueles indivíduos que irão se reproduzir e aqueles que serão descartados. A mais simples é pegar a metade melhor e descartar os outros. Essa forma de seleção tem dois problemas que comprometem sua performance, segundo [2]. O primeiro é que ela não distingue soluções boas das muito boas e o segundo é que ela impede que soluções ruins passem adiante mesmo com chance pequena, o que reduz a diversidade genética do grupo de indivíduos que irão reproduzir.

Uma forma de seleção que resolve as questões acima é a seleção por roleta (*Roulette Wheel Selection*), que será utilizada nesse trabalho. Nesse tipo de seleção, cada indivíduo artificial tem uma chance de ser selecionado proporcional a sua performance na função objetivo.

Um número aleatório entre 0 e 1 é gerado e dependendo do setor que conter esse número, um dos indivíduos é selecionado, como mostra a Figura abaixo

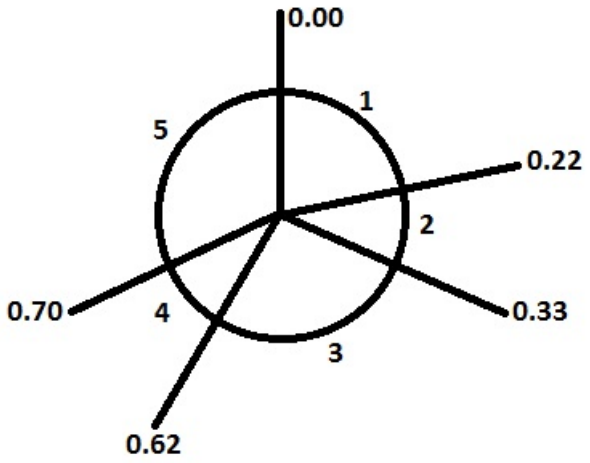

Figura 2.4: Seleção por *Roullete Wheel Selection* para cinco indivíduos. [26]

Dessa forma, quanto melhor for a performance de um indivíduo na geração atual, melhores são as chances de ele ser selecionado para se reproduzir. Além disso, mesmo as soluções ruins tem uma pequena chance de serem selecionadas. Esse é um dos mecanismos que impede o algoritmo de ficar preso a um ótimo local.

A seleção é repetida até que se selecione um numero de indivíduos igual ao tamanho da população atual (O que não significa que a população inteira será selecionada, já que os indivíduos podem ser selecionados mais de uma vez).

2.3.3.3 Recombinação

Terminada a etapa acima, o próximo passo é recombinar os genes dos indivíduos que estão se reproduzindo. Essa etapa é fundamental pois é ela que dá o caráter evolutivo do processo. Nessa etapa, produziremos novas soluções como combinações das anteriores esperando, que essas novas soluções sejam melhores que as anteriores.

Novamente aqui, existem algumas formas diferentes de se fazer isso. Para o nosso problema, utilizaremos a técnica de Recombinação Espalhada (*Scaterred Crossover*). Essa técnica pode ser resumida de seguinte maneira: Um novo vetor binário será gerado aleatoriamente. Esse novo vetor determinará quais genes serão herdados do primeiro indivíduo gerador e quais

serão herdados do segundo indivíduo gerador. Por exemplo, para dois indivíduos

$$[a_1, a_2, a_3, ..., a_n]$$
$$\text{e } [b_1, b_2, b_3, ..., b_n] \tag{2.16}$$

E um vetor gerado

$$[1, 0, 0, ..., 1] \tag{2.17}$$

O novo indivíduo gerado terá a seguinte configuração genética

$$[a_1, b_2, b_3, ..., a_n] \tag{2.18}$$

2.3.4 Mutação

A seguir, ocorre mutação de genes. Essa etapa é importante para diversificar a população e atingir valores de alelos que podem não estar presentes na população inicial e que, de outra forma, não seriam alcançáveis. O processo de mutação em um alfabeto binário é simples, apenas inverte-se o valor do alelo do gene mutado.

Assim como nos outros procedimentos, existem alternativas para a estratégia de mutação. Em nosso trabalho, utilizaremos mutação uniforme com uma probabilidade fixa m. Dessa forma, para cada bit, um número aleatório é gerado entre 0 e 1. Se esse número for menor que m, o valor do alelo é invertido naquele gene.

2.3.5 Nova População

No trabalho original de John Holland [13], a próxima geração deveria ser inteiramente composta pelos descendentes da população anterior. Em [26] se explica que isso seria algo contraproducente sob a perspectiva de otimização. Podemos ter tido um gasto considerável para conseguir uma solução muito boa e nada garante que sua descendente será melhor que ela. Aliás, sob o critério de seleção de roleta, existe a chance, mesmo que muito pequena, da melhor solução não ter sido nem selecionada para reprodução. Dado isso, novos métodos de

composição da nova população, utilizando os conceitos de elitismo e superposição de gerações, devem ser usados para a manutenção da solução ótima.

Elitismo consiste em passar a melhor solução à próxima geração sem alterações. A nova população então seria composta pelo melhor indivíduo da população anterior e por $N-1$ indivíduos gerados na iteração atual. Superposição de Gerações é um passo além, nela é escolhida uma fração g de indivíduos da nova geração. A fração restante $(1-g)$, será composta pelos $(1-g)N$ melhores indivíduos da geração anterior.

2.3.6 Critério de Parada

Como o Algoritmo Genético é um processo iterativo, ele pode, a princípio, ficar funcionando para sempre. Para que isso não aconteça, é necessário estabelecer critérios de parada. Critérios comuns são o número máxima de geração, que estabelece um limite para o número de iterações, tempo máximo, variação a função objetivo menor que um número entre uma iteração e outra e limite de aceitação da solução, critério em que o algoritmo para de funcionar quando uma solução maior (para o caso de maximização) que um limite M for encontrado.

Além desses critérios, existem também os critérios de espera. Esses são utilizados de forma auxiliar aos critérios de parada e existem para que o algoritmo não interrompa seu funcionamento de forma muito prematura e, assim, consiga explorar melhor o espaço de busca. Os critérios de espera usuais são tempo mínimo de execução de algoritmo e número mínimo de iterações.

2.4 Introdução às Metodologias de Predição

Como discutido no primeiro capítulo, ser capaz de estimar a demanda pode aumentar a competitividade de uma organização. Para [14], previsões de vendas, por exemplo, contribuem para a eficiência de setores distintos e dependentes dentro de uma empresa. São, de certo modo, entradas para estratégias de negócios e previsões de recursos de produção.

Nesse trabalho, então, a utilização de algoritmos de previsão se tornaram de extrema importância. Existem diversos tipos de tais algoritmos, como disserta [1], dentre eles: suavização exponencial, modelos autorregressivos (AR), modelos de médias móveis (MA) e a integração

desses dois últimos, modelos ARIMA. Pode-se utilizar, também, elementos de Redes Neurais e Lógica *Fuzzy*.

Como faz parte dos princípios desse capítulo, abordaremos somente os conceitos de relevância ao projeto. À vista disso, no ateremos a caracterização dos algoritmos de suavização exponencial (*exponential smoothing*). Mas, antes, devemos entender um pouco mais sobre séries temporais.

2.4.1 Séries Temporais

Para [28], séries temporais são funções que podem ser observadas ao longo do tempo. Representam valores de ações na bolsa, temperatura de um lago, população em uma cidade e, conforme nosso interesse, a demanda por determinado produto.

Assim, uma série temporal discreta, para [1], pode ser matematicamente expressa como:

$$y_t = \{y_t \in \mathbb{R} \mid i = 1, 2, \cdots, n\} \tag{2.19}$$

onde i é um índice temporal e n o número de observações.

Entende-se, como em [28, 14], que séries temporais podem ser decompostas em parcelas com características especiais. Desse modo, ao analisarmos mais profundamente as dinâmicas de uma função no tempo, poderemos nos deparar com propriedades como:

- Tendência (T): vista como a direção de longo prazo da série;
- Sazonalidade (S): um padrão que se repete com periodicidade conhecida;
- Ciclo (C): padrão repetido com periodicidade não conhecida e mutável;
- Erro (E): Componente não previsível de uma série.

Uma série temporal pode não ser formulada com todas essas parcelas. Elas podem, por outro lado, influenciar a série em soma $T + S + D + E$ ou outras composições.

2.4.2 Modelos de Suavização Exponencial

Historicamente, como descreve [28], modelos de suavização exponencial são utilizados como algoritmos de predição. Sua principal propriedade reside no fato de atribuir pesos a observações passadas. Quanto mais recente a observação, maior o seu peso na previsão de valores

futuros. Em [28], discute-se que o primeiro a elaborar um método com tais características foi Robert G. Brown, analista na marinha americana, em 1944. Seu objetivo era rastrear a velocidade e o ângulo utilizados ao atirar em submarinos.

Existem diversos métodos que se caracterizam por ser uma suavização exponencial, dentre eles, como mostra [28, 1], temos o método da suavização exponencial simples, o método linear de Holt, o método da tendência amortecida e outros.

Nesse trabalho, nosso foco será o método da suavização exponencial simples, por ser mais semelhantes ao que se emprega no projeto. Como é mostrado em [6], o método de suavização exponencial requer baixo armazenamento de dados, é relativamente acurado para curtas previsões e uma aproximação para um controlador de atraso de primeira ordem, utilizado na Teoria de Controle, é rapidamente entendido.

2.4.2.1 Suavização Exponencial Simples

Observados valores passados, incluindo o instante $t-1$, pretendemos prever o próximo valor de uma série, y_t. Assim, feita a previsão $\hat{y}_t$, é possível, em momento oportuno, medir o erro na estimação, $y_t - \hat{y}_t$. É justamente essa a base do método de suavização exponencial simples. De acordo com [28], este é caracterizado pela equação:

$$\hat{y}_{t+1} = \hat{y}_t + \alpha(y_t - \hat{y}_t) \text{ com } 0 \leq \alpha \leq 1 \tag{2.20}$$

Interessante notar que realizando mudanças algébricas, substituindo valores de $\hat{y}_t$ por seu equivalente $\hat{y}_{t-1} + \alpha(y_{t-1} - \hat{y}_{t-1})$ e substituições semelhantes com valores anteriores, entendemos o motivo do nome do método, suavização exponencial. Ao passo que o dado observado se torna antigo, ele é atenuado pelo valor de α^{λ}.

2.4.3 Previsão Indireta da Demanda

Infelizmente, a tentativa de determinação dos valores futuros de uma demanda, como ocorre nesse trabalho, é prejudicada pela falta de capacidade de observação dos valores ocorridos. Isso, pois, para a organização, a demanda é manifestada através do volume de vendas por determinado produto.

O que se dá muitas vezes é a não concretização de todas as vendas requeridas pela demanda no período. Estoque baixo, por exemplo, evita que se atinja a totalidade da procura.

Ocorrências como essas são difíceis de serem contabilizadas, assim a observação da demanda é indireta e imprecisa. Com isso em mente, sabemos que prever valores futuros da procura pelo produto acarretará em possíveis erros. Mas, ainda assim, a previsão, mesmo que indireta, é importante para o sucesso desse projeto.

Capítulo 3

Projeto de Controle

Introduzido o problema e discutidos os tópicos teóricos necessários, já é hora de estudarmos as soluções para o controle do sistema. Como muitos problemas, o projeto pode não ser um processo contínuo e suave, sendo necessário rever as metodologias empregadas rotineiramente. Avanços em um dia são facilmente superados pela obrigação de ter que recomeçar muitas vezes.

Ao fim, os resultados finais acabam por não traduzir ao leitor as dificuldades encontradas ao longo da elaboração e do estudos das soluções. Assim, esse capítulo responsabiliza-se por ilustrar o caminho percorrido durante o projeto, as dificuldades encontradas e os motivos da programação de um simulador em plataforma MATLAB.

Mas, sem dúvidas, a parte essencial deste trabalho, as soluções estudadas, terão um foco especial nesse capítulo.

Isto posto, inicia-se o capítulo com os passos computacionais adotados, quais *softwares* foram utilizados e os motivos pelos quais não se mostraram adequados as nossas necessidades.

Após, estrutura-se nosso modelo como um problema clássico de controle. Podemos, então, melhor compreender as dificuldades encontradas no projeto de estratégias de solução.

Logo na sequência, as técnicas de predição de demanda são apresentadas. Estas, permearam todas as estratégias utilizadas, sendo este o motivo para ganharem uma seção individualizada.

Ao final, porém o mais importante, discorremos sobre os controladores elaborados e estudados, bem como a otimização de seus parâmetros a fim de garantir um EVA máximo.

Enfim, introduzidos os tópicos abordados nesse capítulos, passemos a engenharia da solução.

3.1 Métodos Computacionais Utilizados

O engenheiro moderno, além da bagagem teórica e capacidade analítica, possui uma ferramenta poderosa no auxílio aos seus desafios: o computador. Ao passo que se evoluem as plataformas computacionais, *softwares* especializados e complexos são elaborados e ganham papel importante tanto no meio acadêmico, quanto no industrial. Assim, não lançar mão desses recurso significaria um atraso nos estudos da dinâmica do sistema e na concepção das estratégias de controle e otimização.

Portanto, para que possamos ter êxito em nossa tarefa necessitamos de uma ferramenta computacional que nos permita flexibilidade e facilidade na programação de diversos algoritmos, além de permitir entender (ou escolher) os métodos envolvidos, como a ordem de solução do sistema de equação a diferenças.

O problema é que não se tem todos os requisitos em mente logo ao iniciar o projeto. Isso, necessariamente, acarretou em ciclos de avanços e atrasos no cronograma.

3.1.1 iThink ©

Logo ao ingressar nessa proposta de trabalho, fomos introduzidos ao iThink. Essa é uma ferramenta desenvolvida pela ISEE Systems e possui, como clientes principais, os tomadores de decisão em organizações de negócios. Portanto, tem como essência, a lógica empresarial.

A decisão por começar o projeto com o iThink foi em decorrência da sua utilização em [12], referência na qual baseamos nosso problema.

Trabalhar com esse *software* é, inicialmente, extremamente intuitivo. Não são necessárias equações matemáticas para a elaboração de simulações e estudo de estratégias. Faz-se tudo por meio de um modelo de fluxos ilustrados. As equações são automaticamente extraídas do fluxo. A Figura 3.1 ilustra a forma pela qual se cria modelos no programa.

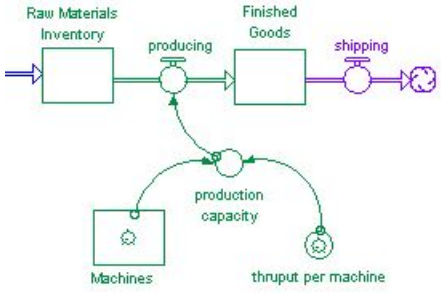

Figura 3.1: Modelo em fluxo - iThink (retirado do *site* do desenvolvedor [33])

Os estudos iniciais da dinâmica do sistema de estoque se mostraram simples de serem feitos. Mudar parâmetros ou outras alterações pontuais também não se mostravam difíceis de serem realizadas. Ao passo que avançamos na complexidade de nossas estratégias, o *software* reduzia sua praticidade. Como não era possível a edição de equações matemáticas, a criação de um simples controlador PID era uma tarefa, no mínimo, desanimadora.

A redução na flexibilidade de elaboração de estratégias mais complexas por si já seria uma razão para pensar em mudar a ferramenta computacional. Mas, também, não era possível checar a existência de possíveis "*loops* algébricos" decorrentes de erros na modelagem. Logo, como de fato ocorreram tais deslizes, os resultados retornados pelas simulações não representavam com exatidão o que pretendíamos. Isso, pois mesmo com a existência de "*loops* algébricos" matemáticos, o *software* era capaz de resolvê-los, porém de uma maneira não simples de ser verificada. Foi necessário um longo período analisando tabelas de resultados no tempo, até ser possível entender a ordenação das equações e solução dos "*loops*".

Ainda, não se podia determinar com certeza se aquela ordenação verificada era a mesma sempre que se rodava a simulação. Assim, por tais fatores, foi necessário deixar o iThink de lado.

Como uma analise da experiência com a plataforma, entende-se que esta é adequada a elaboração de modelos no quais não se procura exatidão matemática e, também, quando não se pretende construir estratégias complexas, como Algoritmos Genéticos, por exemplo.

3.1.2 Berkeley Madonna ©

A migração natural foi para o Berkeley Madonna. Afinal, em [12] também se utiliza essa plataforma. A metodologia encontrada em [12] era a seguinte: criava-se o modelo e estudava-se a dinâmica no iThink e, quando necessário, exportar-se-ia o código ao Berkeley Madonna para a utilização de sua ferramenta de otimização *OPTIMIZE*.

Assim, como nosso projeto envolvia otimização e o próprio iThink não nos era mais adequado, decidimos por adotar o Madonna para a otimização e estruturação da modelagem.

De natureza igual ao iThink quanto a facilidade de utilização, esse *software* é recomendado a estudos da dinâmica de sistemas. É possível, nele, a edição do modelo pelas equações. E, criado o sistema matemático, a utilização da ferramenta *OPTIMIZE* envolvia, somente, determinar quais parâmetros deveriam ser otimização e o intervalo de busca. As restrições do problema de controle ótimo eram formuladas no conjunto de equações.

Com foco em sistemas contínuos, o *software* apresenta, porém, uma incômoda restrição quando se cria modelos discretos. A resolução de equações a diferenças, como o próprio guia do usuário revela [18], é limitada a forma:

$$x(k+1) = f\{x(k)\} \tag{3.1}$$

Assim, algo da forma $I(k) = I(k-1)+O(k-\tau)-S(k)$ não era possível de ser realizada, já que a variável dependente $I(k)$ é uma função de outras variáveis no mesmo instante de tempo k. Portanto, deveríamos alterar o nosso modelo ou a ferramenta computacional. Preferiu-se a segunda alternativa. Isso ocorreu, porém, pois já se sentia a necessidade de utilizar algo que permitisse elaborar estratégias diversas, como controlador *fuzzy*, de maneira simples.

Além, por mais que utilizar *OPTIMIZE* fosse prático, não se sabia ao certo quais metodologias de otimização eram, nele, implementadas. Logo, não poder-se-ia justificar ou questionar seus resultados. Preferiu-se, portanto, alterar novamente a plataforma de simulação.

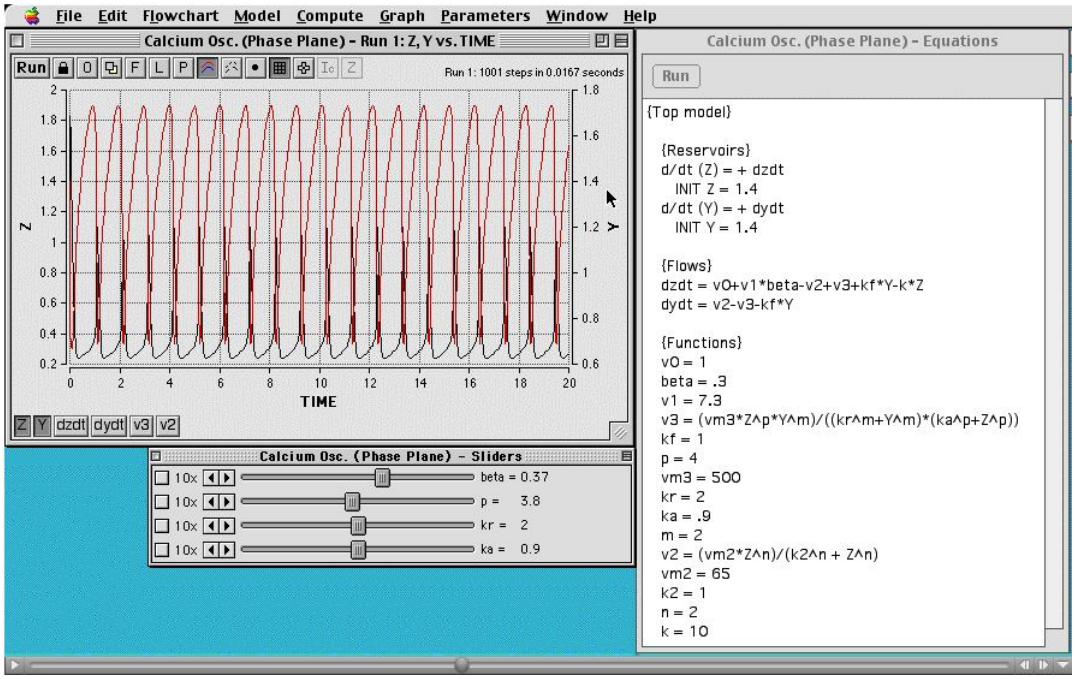

Figura 3.2: Tela gráfica - Berkeley Madonna ©(retirado do *site* do desenvolvedor [19])

3.1.3 Simulink - MATLAB ©

Na realidade, a adoção da *toolbox* Simulink do MATLAB foi efêmera. Logo de início percebeu-se que a construção de um modelo em diagrama de blocos dificultaria a flexibilidade na elaboração de estratégias diversas, aumentando, inclusive, o tempo dispendido na programação do simulador.

À vista disso, passemos a próxima e definitiva ferramenta computacional.

3.1.4 *M-File* - MATLAB ©

A adoção do *M-File* foi, enfim, realizada. O longo processo até que chegasse a esse ponto, sem dúvida, atrasou o cronograma inicial. Mas, há que se admitir, que os momentos iniciais não foram desperdiçados e a experiência com outras ferramentas foi enriquecedora.

Agora, temos a disposição um arsenal de possibilidades com as funcionalidade permitidas pelo MATLAB. E mais, inteira flexibilidade para estruturarmos, da maneira que julgássemos adequadas, a ordenação do sistema de equações.

Outro fator positivo é a existência de "*debug*". Poder checar o passo a passo da simulação, é uma vantagem enorme, tanto para verificações pontuais, quanto para correção de problemas

na programação.

M-Files permitem, também, desfrutar do instrumental voltado ao Algoritmo Genético e à Lógica *Fuzzy*, ambos já elaborados para o MATLAB.

3.1.5 Características do Simulador

O código desenvolvido é extremamente flexível e permite operar simulações diversas. Pode-se escolher entre utilizar o algoritmo genético, utilizar rodadas em bateladas ou simular o sistema para parâmetros únicos.

As nuances do que foram programadas podem ser verificadas ao consultar o código fonte. Nessa seção, será ilustrado o princípio de funcionamento do simulador.

Assim, a Figura 3.3 mostra um diagrama básico sobre como se faz o encaminhamento de dados e a sequência de eventos. O bloco "Algoritmos Genéticos" pode ser habilitado ou não. Caso seja habilitado, ele envia, à rotina de simulação, um vetor de dados $\mathbf{b}$ utilizado para codificar os indivíduos em um vetor de *bits*. Esse vetor é enviado ao Controlador. Esse é responsável por traduzir o vetor para os parâmetros específicos de cada estratégia (por exemplo, K_p e demais parâmetros para um PID).

O controlador é responsável por fornecer o sinal $O(k)$. Aliás, a estratégia pode depender do sinal de demanda prevista $\hat{D}(k)$. Por isso, a caixa Controlador também recebe dados da caixa Preditor. Também é necessário receber dados sobre o vetor estado atual $\mathbf{x}(k)$ para pode calcular os erros em relações as referências de cada variável.

Calculado o sinal de controle, o sistema é executado. Como sistema, entende-se o modelo descrito no capítulo 1 em conjunto a função custo EVA.

Agora, o sinal de vendas $S(k)$ é enviado ao preditor para estimação da demanda futura.

Ao final desse processo, envia-se o valor final do EVA ao bloco de GA.

Caso não se esteja utilizando o GA, basta definir os parâmetros do controlador manualmente.

Se for de interesse do usuário, é possível realizar a exportação para Microsoft Excel e gráficos em geral de quaisquer dados relevantes, bem como gráficos pertinentes ao GA.

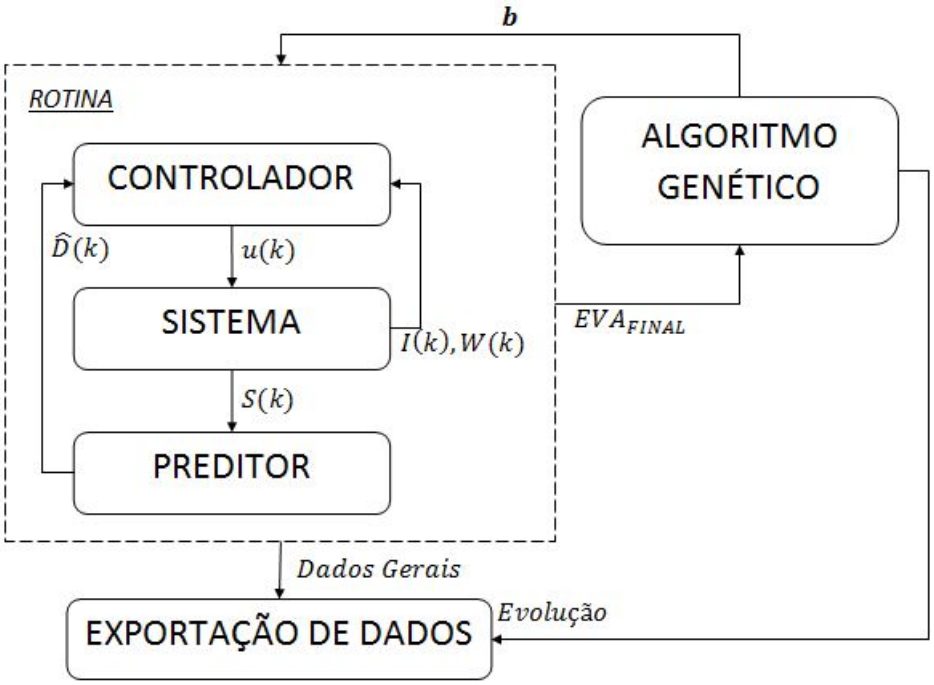

Figura 3.3: Diagrama de funcionamento do Simulador

Para ilustrar o funcionamento da caixa rotina ilustrada na Figura 3.3, o algoritmo 1 descreve o passo a passo programado. Inicia-se com a definição dos parâmetros pertinentes (que podem ser recebidos da caixa GA) e do tipo de controlador e preditor utilizados. Ao final, o EVA é a variável mais importante.

A demanda no instante de tempo k é calculada como retorno de uma função de distribuição *Poisson*. Isso feito, o controlador pode calcular o sinal $O(k)$ em função da estratégia de controle escolhida e da demanda prevista para o momento.

Com o sinal de controle, o sistema (como descrito no sistema de equações do capítulo 1) retorna um vetor **h** de variáveis importantes, como $I(k)$,$W(k)$,$EVA(k)$ e outros.

É possível com $S(k)$ utilizar a função **preditor** para o cálculo da demanda estimada para o instante seguinte.

Algoritmo 1: Rotina de Simulação

Data: Definição de parâmetros, condições iniciais, tipo de controlador e preditor
Result: EVA ao final da simulação
Inicialização;
for $k \leftarrow 1$ **to** 365 **do**
  $D(k) \leftarrow$ ***poisson***(2);
  $O(k) \leftarrow$ ***controlador***$(TipoControle, \hat{D}(k))$;
  $\mathbf{h} \leftarrow$ ***sistema***$(O(k))$;
  $\hat{D}(k+1) \leftarrow$ ***preditor***$(S(k), TipoPreditor)$;
  $k \leftarrow k+1$;
end for

3.2 Maximização do EVA em Horizonte Finito

A estruturação de um modelo de estoque como um problema de controle é, de certa forma, uma metodologia atual [34, 27, 23]. Recapitulando o modelo do sistema, descrito no capítulo 1, temos:

$$\mathbf{x}(k) = \begin{bmatrix} I(k) \\ W(k) \end{bmatrix} \tag{3.2}$$

$$I(k) = I(k-1) + O(k-\tau) - S(k) \tag{3.3}$$

$$W(k) = W(k-1) + O(k) - O(k-\tau) \tag{3.4}$$

Escolhendo a variável $ordering = O(k)$ como controle e interpretando a demanda como uma entrada exógena, podemos introduzir a notação que segue, a fim de escrever a equação no espaço de estados.

$$u(k) = O(k) \tag{3.5}$$

$$L(k) = S(k) = f(D(k)) \tag{3.6}$$

$$R(k) = O(k-\tau) = u(k-\tau) \tag{3.7}$$

$$\mathbf{x}(k) = \mathbf{x}(k-1) + \begin{bmatrix} O(k-\tau) \\ O(k) - O(k-\tau) \end{bmatrix} + \begin{bmatrix} -f(D(k)) \\ 0 \end{bmatrix} \tag{3.8}$$

Caso exploremos mais a equação 3.8, podemos chegar a seguinte equação matricial

$$\mathbf{x}(k) = \mathbf{x}(k-1) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) + \begin{bmatrix} 1 \\ -1 \end{bmatrix} u(k-\tau) + \begin{bmatrix} -1 \\ 0 \end{bmatrix} L(k) \tag{3.9}$$

Nota-se em 3.9, que a dinâmica do estado do sistema depende de valores atuais e atrasados do sinal de controle, além de uma entrada exógena.

Podemos definir o sinal de controle, por realimentação de estados, da seguinte forma

$$O(k) = \varphi(\zeta, \mathbf{x}(k), \hat{D}(k)). \tag{3.10}$$

onde ζ é o vetor de parâmetros do controlador, como ganhos e referências.

Logo, o problema de controle ótimo em horizonte finito é visto por:

$$\begin{aligned} &\max_{\zeta} \; EVA(k_{horizonte}) \\ \text{s.a } &\mathbf{x}(k) = G(\mathbf{x}(k-1), \varphi(\zeta, \mathbf{x}(k), \hat{D}(k)), \tau) + L(k) \end{aligned} \tag{3.11}$$

É importante ressaltar que estamos interessados no valor da função objetivo em um horizonte finito, e não nos preocupa a capacidade de rastreamento dos controladores e a suavidade dos sinais por eles gerados. Além disso, apenas o valor final da função custo é de interesse para a avaliação do desempenho dos controladores.

3.3 Sistemas de Predição

Como foi dito no capítulo 1, a estimação do volume de vendas futuras é de extrema importância para o gestor de estoque. Prever quantos produtos serão comprados ajuda a determinar melhor a quantidade que deve ser pedida. Logo, o preditor de demanda exerce influência direta na malha de realimentação e, por consequência, na estratégia de controle.

Não custa lembrar que não podemos estimar a demanda diretamente. Na realidade, o sinal estimado é uma função do volume de vendas. O gestor não pode mensurar precisamente a demanda por seus produtos em determinado momento, ele só é capaz de medi-la a partir da quantidade de produtos que foram adquiridas. Assim, os preditores desenvolvidos serão filtros do sinal de vendas e não da demanda.

Nesse trabalho, adotaremos dois modelos de preditores. Ambos possuem caráter semelhante. O primeiro é um filtro de segunda ordem do sinal de vendas, o segundo um preditor por suavização exponencial clássico.

3.3.1 Preditor 1 - Filtro de Segunda Ordem

O preditor 1 é matematicamente expresso como

$$\hat{D}(k+1) = 2S(k) - S(k-1) \tag{3.12}$$

O sinal do preditor é inferiormente saturado em zero. Uma demanda estimada negativa não possui significado objetivo. Assim, o modelo é melhor expresso por

$$\hat{D}(k+1) = \begin{cases} 2S(k) - S(k-1) & \text{se } 2S(k) - S(k-1) > 0 \\ 0 & \text{caso contrário} \end{cases} \tag{3.13}$$

Interessante observar que mesmo não sendo caracterizado matematicamente como um preditor de suavização exponencial de acordo com a equação 2.20, é possível notar que termos passados reduzem sua influência ao preditor com o passar do tempo. Essa é a característica fundamental ao método exposto na seção 2.4.2.1. A partir de 3.12, podemos perceber que

$$\begin{aligned} \hat{D}(k) &= 2S(k-1) - S(k-2) \\ \text{Assim } S(k-1) &= \frac{1}{2}\left\{\hat{D}(k) + S(k-2)\right\} \end{aligned} \tag{3.14}$$

Logo, com 3.14 em 3.12, tería-se

$$\hat{D}(k+1) = S(k) - \frac{1}{2}\left\{\hat{D}(k) + S(k-2)\right\} \tag{3.15}$$

Caso continuássemos as substituições, perceberíamos que as parcelas antigas do preditor, assim como do sinal de vendas, são atenuadas por exponenciais do tipo $(\frac{1}{2})^{\lambda}$

Para ilustrar o desempenho do Preditor 1, faremos com que ele estime um sinal com distribuição *poisson* com média em dois. Para cem iterações, o sinal predito é ilustrado na Figura 3.4

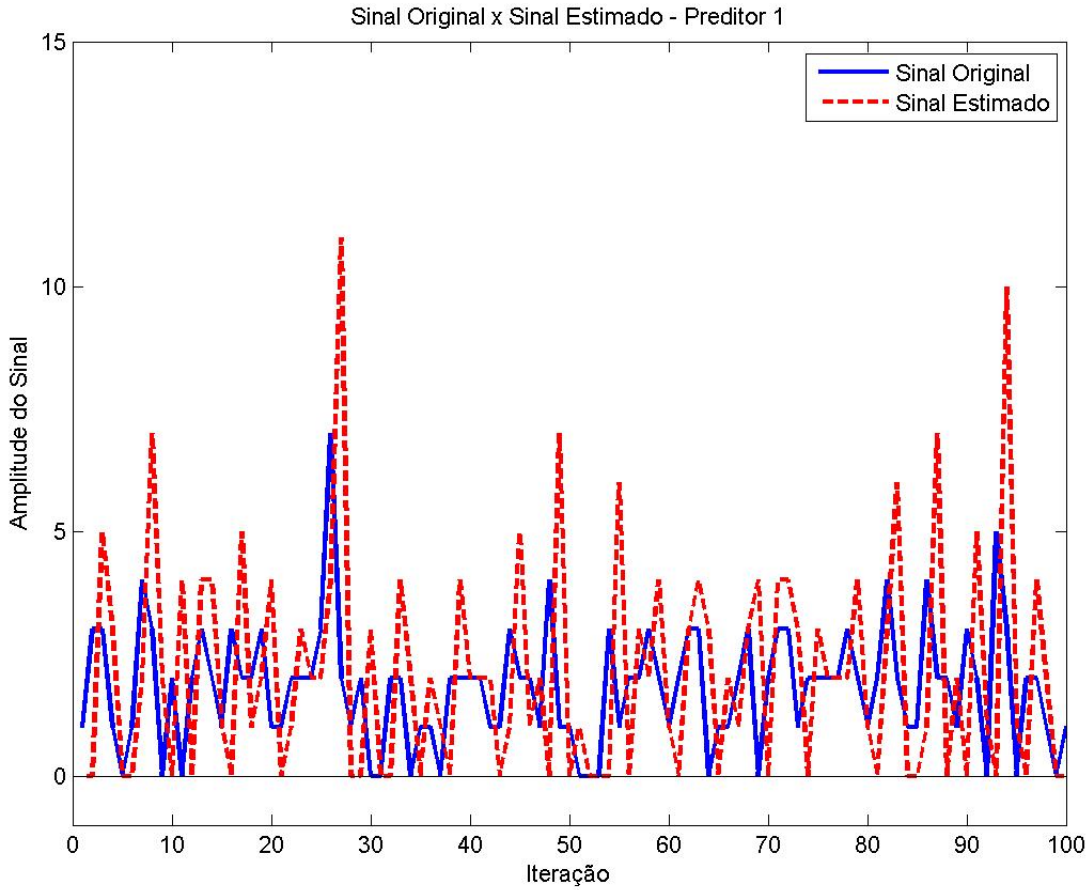

Figura 3.4: Sinal Original e Sinal Estimado - Preditor 1

Nota-se, pela Figura 3.4, que o sinal estimado possui aparência semelhante ao original. Porém, os picos desse sinal são significativamente maiores do que os do original.

Não é necessário, para o preditor 1, utilizar a função de aproximação para inteiro, já que o valor estimado é uma função de uma variável essencialmente inteira, não podendo, esta, assumir valores fracionários ou irracionais.

3.3.2 Preditor 2 - Preditor de Suavização Exponencial Simples

O segundo preditor segue o modelo matemático expresso em 2.20. Assim, o modelo completo é descrito por

$$\hat{D}(k+1) = \begin{cases} h = \hat{D}(k) + \textbf{round}(0.9\left\{S(k) - \hat{D}(k)\right\}) & \text{se } h > 0 \\ 0 & \text{caso contrário} \end{cases} \tag{3.16}$$

A equação 3.16 é a mesma que descreve o modelo de predição com exponencial simples, para $\alpha = 0.9$. Graficamente, o desempenho pode ser ilustrado pela Figura 3.5

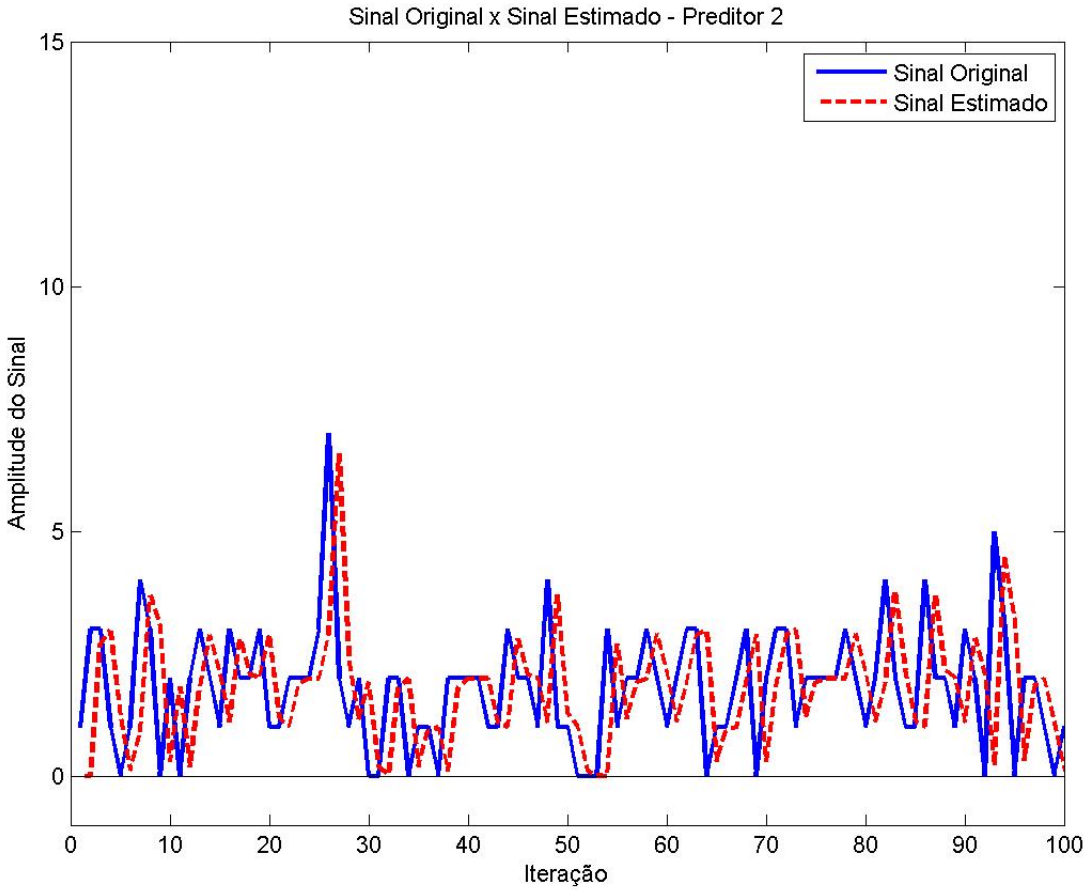

Figura 3.5: Sinal Original e Sinal Estimado - Preditor 2

Nota-se, agora, um sinal estimado de menor amplitude, porém com características dinâmicas aproximadas às do sinal original, embora com atraso aproximadamente constante, visível na defasagem dos picos na Figura 3.5.

3.4 Estratégias de Controle

De maneira geral, os controladores adotados utilizam informações da planta, como o nível atual em estoque $I(k)$, as referências (ou *setpoints*) e os ganhos. Assim, o diagrama da Figura 3.6 ilustra um modelo genérico para os controladores estudados.

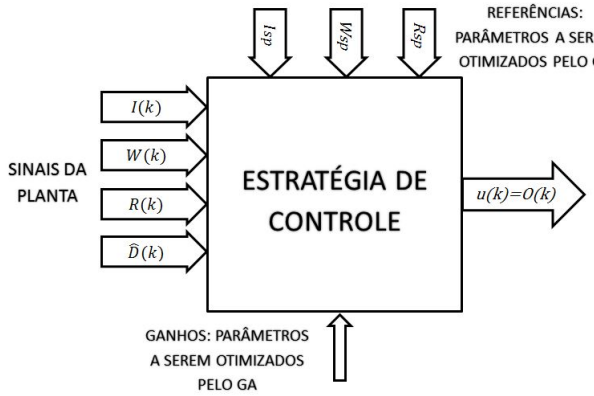

Figura 3.6: Estrutura genérica para os controladores estudados

Importante ressaltar que algumas estratégias não utilizam todos os sinais da planta ilustrados na Figura 3.6 e/ou possuem diferentes conjuntos de referências e ganhos.

3.4.1 Controlador 1 - Modelo Hannon/McGarvey

O primeiro controlador que utilizaremos é controlador proposto por [12], que é similar ao controlador proposto por [7]. Ou seja, basicamente dois controladores P acoplados que, somados à previsão de demanda para o instante atual, fornecerão a saída de controle. Matematicamente, é assim descrito

$$u(k) = \hat{D}(k) - \textbf{round}\{K_{p1}(I(k) - I_{sp}) + K_{p2}(W(k) - W_{sp})\} \tag{3.17}$$

O ajuste do controlador se dá pelos ganhos de realimentação K_{p1} e K_{p2} (denominados $ReactionStore$ e $ReactionShipped$, respectivamente, em [12]) e pelos valores de referência I_{sp} e W_{sp} (denominados $StoreTarget$ e $ShippedTarget$, respectivamente, em [12]). Desse controlador podemos retirar duas informações: a primeira é que ele sempre agirá de forma reativa, ou seja, quanto mais os níveis das variáveis de estado estiverem diferentes dos valores de referência estabelecidos, maior será a ação de controle. Além disso, o controlador não possui memória, nem tenta antecipar variações futuras de $I(k)$ ou $W(k)$.

A parte não-linear do controlador se dá pelas funções **round** (aproximação para inteiro) e pela limitação inferior em zero, ambas utilizadas para respeitar as restrições do problema. A função **round** é utilizada para que o número de peças da encomenda seja um número inteiro,

pois não faz sentido para esse problema um número de natureza diferente para a saída de controle (não é possível encomendar frações de peça). A limitação inferior em zero, por sua vez, é utilizada para que não se encomende números negativos de peça, o que corresponderia a pegar uma peça estocada e retornar para o fornecedor.

Interessante notar que a equação 3.17 descreve um controlador de três entradas e uma saída. A primeira entrada se refere ao preditor de demanda, a segunda e terceira entradas descrevem os erros $e_I(k) = I(k) - I_{sp}$ e $e_W(k) = I(k) - I_{sp}$, respectivamente. Desse modo, o controlador Hannon e McGarvey pode ser ilustrado pela Figura 3.7

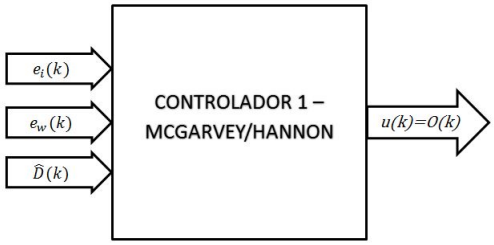

Figura 3.7: Estrutura do Controlador 1 - Hannon/McGarvey

3.4.2 Controlador 2 - Modelo Tosetti

Dois dos controladores que utilizaremos são da forma PID. Começando com um modelo utilizado também por [34]. Esta, deriva-se de uma versão mais simples da discretização do controle PID contínuo pelo método *Backward Euler*.

O controlador PID utilizado aqui é

$$\begin{aligned} u(k) - u(k-1) &= \textbf{round}\{K_p[e(k) - e(k-1)] \\ &+ K_i e(k) + K_d[e(k) - 2e(k-1) + e(k-2)]\} \end{aligned} \tag{3.18}$$

É possível fazer

$$\begin{aligned} u(k) &= u(k-1) + \textbf{round}\{K_p[e(k) - e(k-1)] \\ &+ K_i e(k) \\ &+ K_d[e(k) - 2e(k-1) + e(k-2)]\} \end{aligned} \tag{3.19}$$

Lembrando que o sinal de controle é designado pelo volume pedido $O(k)$ e que este deve ser limitado inferiormente em zero.

O erro é calculado de maneira semelhante ao encontrado no trabalho de [34], logo, matematicamente é expresso por

$$e(k) = [I_{sp} - I(k-1)] + [\hat{D}(k) - W(k-1)] \tag{3.20}$$

Ao contrário do que ocorre no Controlador 1, o erro aqui é um acoplamento entre a diferença do volume atual de estoque e a referência, somada a diferença entre a previsão de demanda e o que estiver em trânsito. Assim, a demanda prevista funciona como uma referência para o volume de produtos sendo processados ($W(k)$). Dessa forma, o controlador Tosetti pode ser compreendido com sendo de uma entrada e uma saída, como ilustra a Figura 3.8

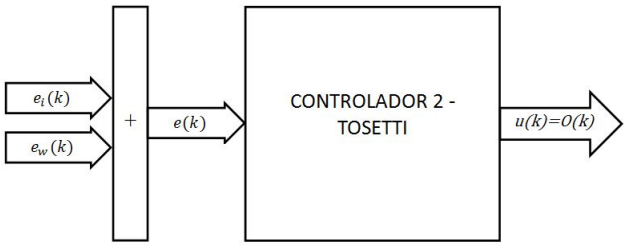

Figura 3.8: Estrutura do Controlador 2 - Tosetti

3.4.3 Controlador 3 - PID Discreto

Além do PID descrito na seção 3.4.2, testaremos uma variante discreta que possua uma associação direta com o PID contínuo. Nesse Controlador, a parte proporcional representa uma reação ao estado atual, a parte derivativa é uma forma de incorporar ao controlador a variação atual e a parte integral funciona utilizando todo o erro acumulado até o instante atual do sistema. Nesse sentido, fazendo com que as funções se mantenham, podemos imaginar um controlador, em sua forma mais simples, como

$$u(k) = u(k-1) + \textbf{round}\{K_p e(k) + K_i \sum_{i=1}^{k} e(i) + K_d[e(k) - e(k-1)]\} \tag{3.21}$$

Lembrando que o sinal de controle é designado pelo volume pedido $O(k)$ e que este deve ser limitado inferiormente em zero. O erro é calculado da mesma forma que na equação 3.20.

Analogamente ao controlador 2 (Tosetti), a equação 3.20 descreve um controlador de uma entrada e uma saída. Assim, o controlador 3 pode ser ilustrado pela Figura 3.9.

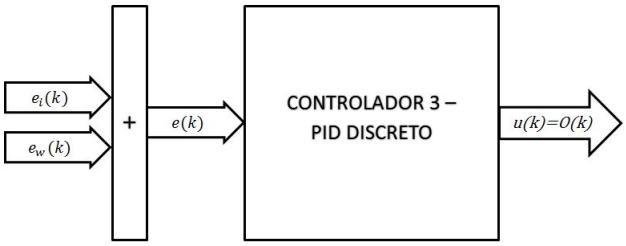

Figura 3.9: Estrutura do Controlador 3 - PID Discreto

3.4.4 Controlador 4 - Realimentação *Receiving*

O próximo controlador que testaremos nesse trabalho é similar ao proposto por McGarvey e Hannon. A diferença consiste na realimentação tripla ao invés de dupla, pois a grandeza $receiving(k) = O(k - \tau)$, que representa a quantidade de itens que chegou em determinado dia, também possuirá um valor de referência e um ganho associado. Esse controlador tem como objetivo aumentar o controle sobre o estoque e evitar grandes variações do mesmo, pois utiliza, não apenas a quantidade em trânsito (que pode chegar no próximo instante de tempo ou mesmo depois de τ (*lead time*) instantes de tempo), mas também a quantidade efetivamente recebida naquele instante de tempo. Matematicamente, funciona da seguinte maneira:

$$u(k) = \hat{D}(k) - \textbf{round}\{K_{p1}(I(k) - I_{sp}) + K_{p2}(W(k) - W_{sp}) + K_{p3}.(R(k) - R_{sp})\} \quad (3.22)$$

Nesse controlador, o ajuste se dá em 6 parâmetros: K_{p1}, K_{p2} e K_{p3} (que seguindo a lógica de [12] podem ser descritos por $ReactionStore$, $ReactionShipped$ e $ReactionReceiving$, respectivamente) além de I_{sp}, W_{sp} e R_{sp} (novamente seguindo a nomenclatura de [12], as referências podem ser denominadas $StoreTarget$, $ShippedTarget$ e $ReceivingTarget$, respectivamente). As funções não lineares nesse controlador possuem o mesmo efeito do controlador proposto por Garvey-Hannon: que a saída de controle respeite as restrições impostas pelo problema.

Pelo aumento no número de sinais de erro, com a inclusão de $e_R(k) = R(k) - R_sp$, o controlador 4 é do tipo quatro entradas e uma saída. Assim, é ilustrado pela Figura 3.10.

Figura 3.10: Estrutura do Controlador 4 - Realimentação Receiving

3.4.5 Controlador 5 - Controlador Fuzzy

A elaboração de uma estratégia *fuzzy* foi possível após um período de experiências com a dinâmica do sistema estudado. Entendendo como este respondia as diversas excitações foi possível o projeto no domínio nebuloso.

Desse modo, seguindo a sequência necessária para a implementação de um controlador *fuzzy* , discutida na seção 2.2, devemos iniciar com a definição das entradas e saídas do controlador.

Como entradas, iremos utilizar os desvios entre as referências para o estoque e volume sendo processado e o estado atual do sistema. Assim, as entradas são definidas pelo conjunto de equações

$$e_1(k) = e_i(k) \quad = \quad I_{sp} - I(k) \tag{3.23}$$

$$e_2(k) = e_w(k) \quad = \quad W_{sp} - W(k) \tag{3.24}$$

A equação 3.23 descreve a amplitude da diferença entre o que desejamos para o estoque e o volume atual. A equação 3.24 é análoga para o *Work-in-Progress* .

Em seguida, devemos definir quais serão as saídas de nosso controlador. Aqui, tem-se apenas um sinal a ser calculado, o $O(k)$. Logo, esse pode ser modelado como uma transformação dos sinais de entrada pela função T. Assim

$$O(k) = T\{e_1(k), e_2(k)\} \tag{3.25}$$

Com entradas e saídas definidas, podemos descrever o controlador *fuzzy* como sendo do tipo duas entradas e uma saída, como ilustra a Figura 3.11.

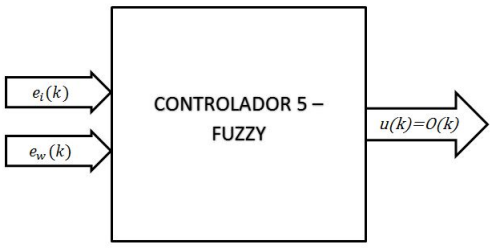

Figura 3.11: Estrutura do Controlador 5 - *Fuzzy*

É, agora, necessário definir os universos de discurso para cada variável do problema. Assim, ambas as entradas serão definidas para o intervalo

$$\mathcal{U}_{\infty,\in} = [-200, 150] \tag{3.26}$$

O intervalo definido em 3.26 é o viável para o problema. Define-se que o erro não é maior do que 150 e menor que -200. Isso, pois existe uma saturação superior em 200 e inferior em 0, no nível de estoque. Além disso, a referência para o mesmo não ultrapassa 127 (escolha de projeto). Análise análoga é feita ao *Work-in-Progress* , com exceção a saturação superior. Essa não é necessária, devido a ação do próprio controlador.

Para a saída, o universo de discurso é definido em

$$\mathcal{Y} = [0, 15] \tag{3.27}$$

O controlador fuzzy exige que se defina o intervalo de saída, assim, como decisão de projeto, criou-se uma saturação superior em 15. Isso significa um produto a mais que o a média da demanda (2) multiplicada pelo tempo de entrada (*lead time*). Além disso, a análise da dinâmica dos outros controladores levou a julgar essa, uma boa opção.

Passando aos valores linguísticos, a entrada e_1 será caracterizada pelo conjunto

$$\begin{aligned} \tilde{A}_1^j &= \{\text{“}\textit{StoreMuitoAlto}''\text{, “}\textit{StoreAlto}''\text{, “}\textit{StorePoucoAlto}'' \\ &\quad \text{, “}\textit{Ideal}''\text{, “}\textit{StoreBaixo}''\text{, “}\textit{StoreMuitoBaixo}'' : j = 1, 2, \ldots, 6\} \end{aligned} \tag{3.28}$$

Já a entrada e_2 será linguisticamente definida pelo conjunto

$$\begin{aligned}\tilde{A}_2^j &= \{\text{“}\textit{ShippedMuitoAlto}\text{”}, \text{“}\textit{ShippedAlto}\text{”}, \text{“}\textit{ShippedPoucoAlto}\text{”} \\ &\quad , \text{“}\textit{Ideal}\text{”}, \text{“}\textit{ShippedBaixo}\text{”}, \text{“}\textit{ShippedMuitoBaixo}\text{”} : j = 1, 2, \ldots, 6\}\end{aligned} \tag{3.29}$$

A saída é expressa pelo conjunto

$$\begin{aligned}\tilde{B}_1^j &= \{\text{“}\textit{MuitoAlto}\text{”}, \text{“}\textit{Alto}\text{”}, \text{“}\textit{Medio}\text{”} \\ &\quad , \text{“}\textit{Baixo}\text{”}, \text{“}\textit{Nulo}\text{”} : j = 1, 2, \ldots, 5\}\end{aligned} \tag{3.30}$$

Agora que temos os conjuntos de valores linguísticos, é possível expressá-los por meio de funções de pertinência. Sendo assim, as Figuras 3.12 e 3.13 ilustram os graus de pertinência para as funções das entradas $e_1(k)$ e $e_2(k)$, respectivamente. A Figura 3.14, por sua vez, faz o mesmo para a saída $O(k)$.

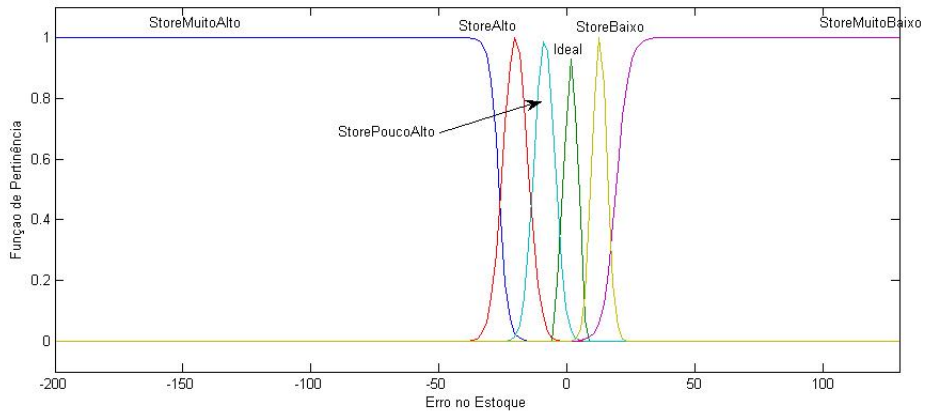

Figura 3.12: Funções de Pertinência para o Erro do Store - $I_{sp} - I(k)$

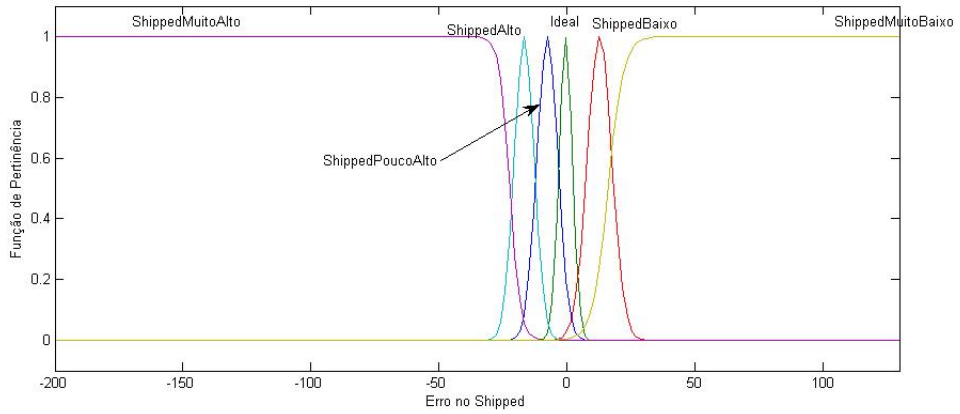

Figura 3.13: Funções de Pertinência para o Erro do Shipped - $W_{sp} - W(k)$

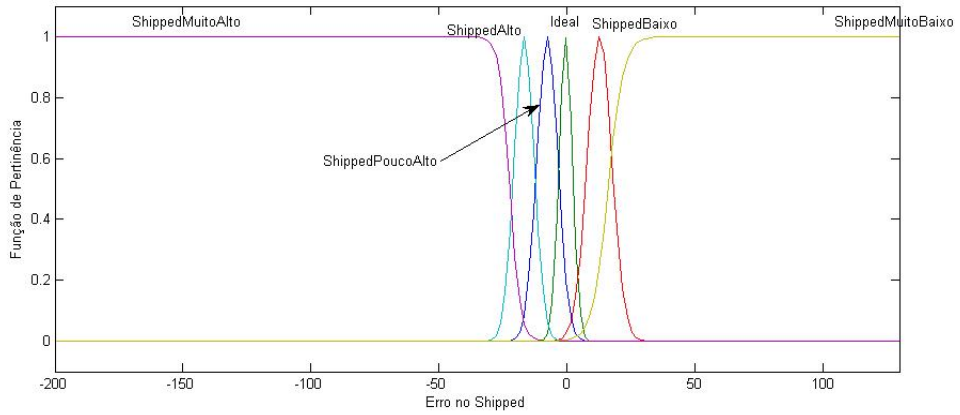

Figura 3.14: Funções de Pertinência para o Sinal de Controle $u(k) = O(k)$

Prosseguindo com o projeto do controlador, é necessário realizar o conjunto de regras de inferência de acordo com a proposição SE-ENTÃO. A tabela 3.1 ilustra as regras utilizadas. É importante ressaltar que essas são baseadas na observação da natureza do sistemas, sendo, portanto, baseadas em experiência e “tentativa e erro”.

Tabela 3.1: Regras de Inferência

Regra	Se(e_1) é	e (e_2) é	Então ($O(k)$) é
1	*StoreMuitoAlto*	-	*Nulo*
2	*Ideal*	*Ideal*	*Nulo*
3	*Ideal*	*ShippedPoucoAlto*	*Baixo*
4	*Ideal*	*ShippedBaixo*	*Baixo*
5	*Ideal*	*ShippedAlto*	*Nulo*
6	*Ideal*	*ShippedMuitoAlto*	*Nulo*
7	*Ideal*	*ShippedMuitoBaixo*	*Baixo*
8	*StoreAlto*	-	*Nulo*
9	*StorePoucoAlto*	*ShippedPoucoAlto*	*Baixo*
10	*StorePoucoAlto*	*Ideal*	*Nulo*
11	*StorePoucoAlto*	*ShippedBaixo*	*Medio*
12	*StorePoucoAlto*	*ShippedAlto*	*Baixo*
13	*StorePoucoAlto*	*ShippedMuitoAlto*	*Nulo*
14	*StorePoucoAlto*	*ShippedMuitoBaixo*	*Baixo*
15	*StoreMuitoBaixo*	*ShippedPoucoAlto*	*Medio*
16	*StoreMuitoBaixo*	*Ideal*	*Medio*
17	*StoreMuitoBaixo*	*ShippedBaixo*	*Medio*
18	*StoreMuitoBaixo*	*ShippedMuitoBaixo*	*MuitoAlto*
19	*StoreMuitoBaixo*	*ShippedAlto*	*Baixo*
20	*StoreMuitoBaixo*	*ShippedMuitoAlto*	*Nulo*
21	*StoreBaixo*	*ShippedPoucoAlto*	*Baixo*
22	*StoreBaixo*	*Ideal*	*Baixo*
23	*StoreBaixo*	*ShippedBaixo*	*Medio*
24	*StoreBaixo*	*ShippedAlto*	*Baixo*
25	*StoreBaixo*	*ShippedMuitoAlto*	*Nulo*
26	*StoreBaixo*	*ShippedMuitoBaixo*	*MuitoAlto*

A partir das regras criadas e dos universos de discursos, é possível gerar uma superfície de controle para a saída $O(k)$ em função do plano caracterizado pelos domínios das entradas.

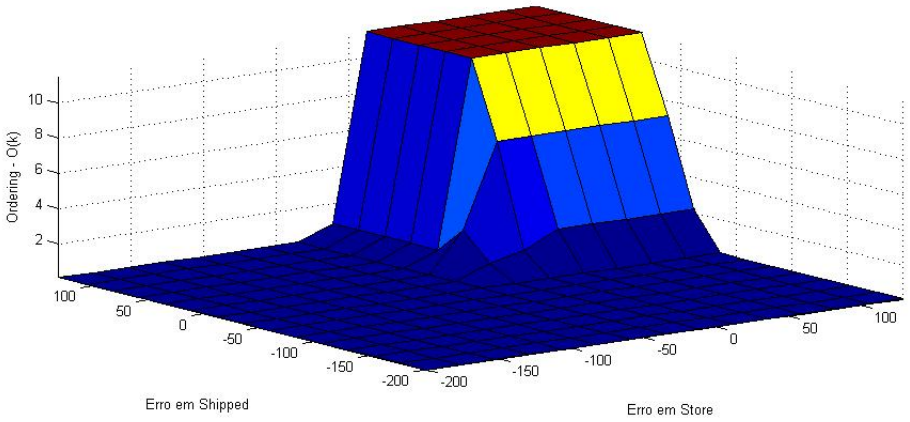

Figura 3.15: Superfície de Controle

Nota-se, na Figura 3.15, que o poder de ação do controlador se resume, principalmente, a quando as variáveis de estado $I(k)$ e $W(k)$ possuem valores abaixo das respectivas referências. Isso é em consequência da não estaticidade dos ganhos do controlador *fuzzy* . Ou seja, o controlador adapta seus ganhos de acordo com a magnitude de suas entradas.

Calculado o valor fuzzy para o sinal de controle, é necessário que este passe pelo processo de Defuzzificação. O método escolhido é o centro de gravidade, expresso pela equação 2.14.

Capítulo 4

Simulações e Resultados

Esse capítulo apresentará os resultados obtidos das simulações e dos valores ótimos para o uso de cada controlador anteriormente apresentado.

Para tanto, esse capítulo se dividirá em três partes principais. Em primeiro lugar será apresentada a metodologia de obtenção de resultados e os ajustes feitos no algoritmo genético por meio do *Global Optimization Toolbox* do MATLAB.

A seguir serão apresentados os resultados em si, bem como a representação gráfica de todas as variáveis relevantes para o entendimento e avaliação dos mesmos. Essa parte do capítulo será dividida em cinco subseções, cada uma delas relativa aos resultados obtidos por cada controlador.

Por fim, será feita a análise dos resultados, bem como a comparação entre os mesmos, para que se saiba qual dos controladores propostos é o mais indicado para ser utilizado no sistema. Nessa seção, além de discutir a performance medida pelos controladores, também comentaremos características relevantes, como o efeito *bullwhip* de cada controlador.

4.1 Metodologia de Simulação e Obtenção de Resultados

Para realizar uma comparação justa entre os controladores, faz-se necessário definir uma metodologia de teste padrão.

Nesse sentido, ainda é necessário explicar as etapas da simulação às quais todos os controladores são submetidos até a obtenção dos resultados.

4.1.1 Método de Extração do Valor Ótimo

Cada controlador é submetido a uma série de procedimentos a fim de atingir o melhor desempenho possível. Assim sendo, o sistema como um todo é submetido a uma série de execuções utilizando o algoritmo genético. O número de rodadas do algoritmo ao qual o sistema será submetido pode ser arbitrado pelo operador do simulador, sendo para esse trabalho utilizado o valor de 10 rodadas do método. Para as execuções do GA, já é adotado a demanda aleatória.

Após esse passo, o melhor resultado obtido entre as 10 rodadas é armazenado, bem como os parâmetros do Controlador que levaram a esse resultado.

O próximo passo será aplicar os parâmetros do controlador obtidos pelo GA a uma série de simulações do sistema, novamente, com demanda variável. Daí poderemos extrair o melhor resultado obtido (melhor EVA) para esse controlador, o resultado médio (dos EVA) e o desvio padrão.

Na seção de apresentação dos resultados serão mostradas todas essas grandezas e representações gráficas detalhadas do melhor resultado obtido por cada controlador.

Em todos os testes, o atraso τ é de 7 dias e a média λ da demanda $D(k)$ é igual a 2, como proposto em [12]. Além disso, o estoque $I(k)$ possui limite superior em 200 unidades.

Nota 4.1 *Foi inicialmente testada a utilização de demanda fixa na média da distribuição de Poisson para as execuções do GA. Quando utilizados os parâmetros extraídos da melhor dessas execuções e simulando o sistema, o resultado é inferior ao obtido quando a demanda aleatória já é adotada no GA.*

4.1.2 Ajustes no Algoritmo

Tendo por objetivo novamente a coerência entre os resultados, também é necessário que se use o algoritmo genético da mesma forma em todas as rodadas de simulação do Sistema. Dessa forma, é também necessário que se padronize o modo de ação do Algoritmo Genético, por meio do *Global Optimization Toolbox*.

Nesse trabalho, uma série de parâmetros do algoritmo serão padronizados, a começar pelo tipo de população, que será o vetor de bits (*Bit-String*). Essa é a forma de assegurar que as

variáveis respeitem suas restrições, isto é, que as variáveis que apenas podem ser representadas por números naturais apareçam dessa forma. A quantidade de indivíduos da população será 20 e a população inicial será obtida de forma aleatória uniforme, ou seja, qualquer ponto dentro do espaço de busca tem a mesma probabilidade de ser escolhido.

A codificação do vetor de bits para os parâmetros do controlador utiliza 8 bits por parâmetro. Dessa forma, para um Controlador que tenha n parâmetros a otimizar, o número de bits utilizado no vetor gerado será 8n. Além disso, os parâmetros são decodificados da seguinte forma:

$$\begin{cases} \mathbf{p}_1 = \frac{\mathbf{b}_1 2^0 + \mathbf{b}_{n+1} 2^1 + \mathbf{b}_{2n+1} 2^2 + \cdots + \mathbf{b}_{7n+1} 2^7}{2^{k_1}} \\ \mathbf{p}_2 = \frac{\mathbf{b}_2 2^0 + \mathbf{b}_{n+2} 2^1 + \mathbf{b}_{2n+2} 2^2 + \cdots + \mathbf{b}_{7n+2} 2^7}{2^{k_2}} \\ \vdots \\ \mathbf{p}_n = \frac{\mathbf{b}_n 2^0 + \mathbf{b}_{n+n} 2^1 + \mathbf{b}_{2n+n} 2^2 + \cdots + \mathbf{b}_{7n+n} 2^7}{2^{k_n}} \end{cases} \tag{4.1}$$

Sendo $\mathbf{p}_i$ é o parâmetro i de otimização, $\mathbf{b}$ é o vetor de bits e k_i é o fator de precisão que nos permite ajustar o intervalo de interesse do parâmetro bem como impor que esse parâmetro seja um número natural, caso aplicável. Para todo parâmetro $\mathbf{p}_j$ que o problema restringe como número natural, k_j é igual a 0. Esse esquema de embaralhar os bits adjacentes entre as variáveis será vantajoso para que o método de recombinação utilizado pelo algoritmo possa atuar sobre todas as variáveis de uma vez ao invés de atuar apenas em uma ou duas, que seria o caso se as variáveis fossem decodificadas sequencialmente, ou seja, os bits de 1 a 8 representariam $\mathbf{p}_1$, 9 a 16 representariam $\mathbf{p}_2$ e assim sucessivamente.

Vejamos o exemplo do Controlador Garvey-Hannon. Nesse Controlador, são quatro os parâmetros de interesse ($n = 4$) a serem otimizados: I_{sp}, W_{sp}, K_{p1} e K_{p2}:

$$\begin{cases} I_{sp} = \frac{\mathbf{b}_1 2^0 + \mathbf{b}_5 2^1 + \mathbf{b}_9 2^2 + \cdots + \mathbf{b}_{29} 2^7}{2^0} \\ W_{sp} = \frac{\mathbf{b}_2 2^0 + \mathbf{b}_6 2^1 + \mathbf{b}_{10} 2^2 + \cdots + \mathbf{b}_{30} 2^7}{2^0} \\ K_{p1} = \frac{\mathbf{b}_3 2^0 + \mathbf{b}_7 2^1 + \mathbf{b}_{11} 2^2 + \cdots + \mathbf{b}_{31} 2^7}{2^5} \\ K_{p2} = \frac{\mathbf{b}_4 2^0 + \mathbf{b}_8 2^1 + \mathbf{b}_{12} 2^2 + \cdots + \mathbf{b}_{32} 2^7}{2^5} \end{cases} \tag{4.2}$$

Nesse exemplo fica claro que as variáveis I_{sp} e W_{sp} são restritas a números naturais, como era de se esperar, visto que $k_1 = k_2 = 0$.

Como enunciamos anteriormente no Capítulo 2, os métodos de seleção, recombinação, mutação, elitismo e os critérios de parada também devem ser definidos. Em nosso trabalho,

utilizaremos seleção por roleta, recombinação *Scattered*, mutação uniforme com probabilidade de 0,01 e elitismo de 2 indivíduos. O critério de parada único será o número de iterações do algoritmo, fixado em 20.

4.2 Apresentação dos Resultados

Para todas as estratégias de controle a seguir, apresentaremos os parâmetros resultantes de cada rodada do GA, bem como valor de EVA associado a eles.

A seguir, para os parâmetros que produziram o melhor EVA anteriormente, faremos uma nova série de 10 simulações. Para o melhor resultado obtido com essa série, apresentaremos todas as variáveis relevantes do sistema, como as variáveis de estado, a saída de Controle, a demanda, sua previsão e as vendas e ainda a evolução diária do EVA.

Também apresentaremos estatísticas relativas a essas 10 simulações utilizando os mesmos parâmetros ótimos obtidos pelo Algoritmo Genético. Apresentaremos a performance média de cada estratégia de controle e o desvio padrão associado, que mensura o quanto essa estratégia de controle se desvia da média, o que, em termos financeiros, tem relação estreita com o risco associado a utilização de tal estratégia.

4.2.1 Controlador Garvey-Hannon

4.2.1.1 Preditor de demanda baseado em séries temporais

Tabela 4.1: Parâmetros obtidos pelo GA em dez simulações para o controlador Garvey-Hannon com demanda estimada por séries temporais

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	20316.42	20989.21	20327.63	20495.64	21111.19	21668.25	21518.15	21474.54	**21778.31**	20524.17
I_{sp}	37	13	31	12	24	1	21	25	**21**	11
W_{sp}	1	19	0	17	0	28	9	4	**4**	21
K_{p1}	0.63	2.78	3.69	6.50	5.25	4.13	5.34	5.66	**4.81**	3.97
K_{p2}	7.56	2.28	3.66	4.81	2.19	3.56	5.38	5.34	**3.69**	2.81

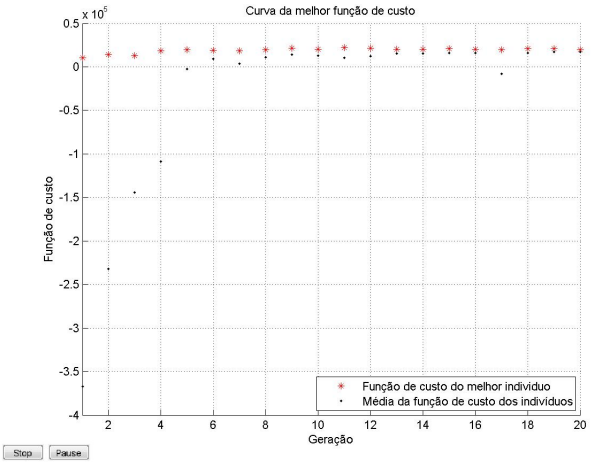

Figura 4.1: Evolução do Algoritmo Genético para o Controlador Garvey-Hannon com demanda estimada por séries temporais

Tabela 4.2: Parâmetros e resultados para o controlador Garvey-Hannon com demanda estimada por séries temporais

Controlador 1	Valor
I_{sp}	21
W_{sp}	4
K_{p1}	4.81
K_{p2}	3.69
Melhor EVA	20437.27
Média EVA	18811.50
Desvio Padrão EVA	1799.95

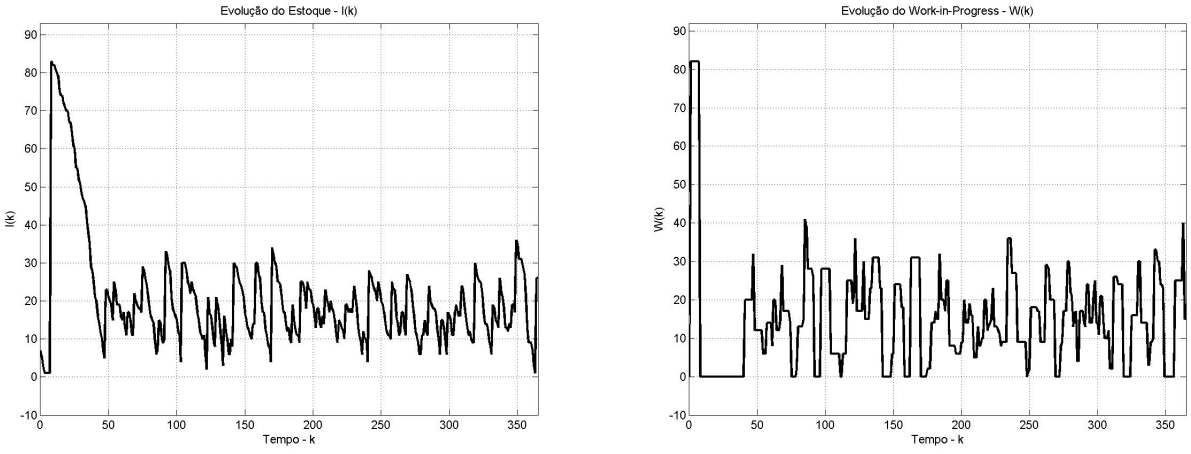

Figura 4.2: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais

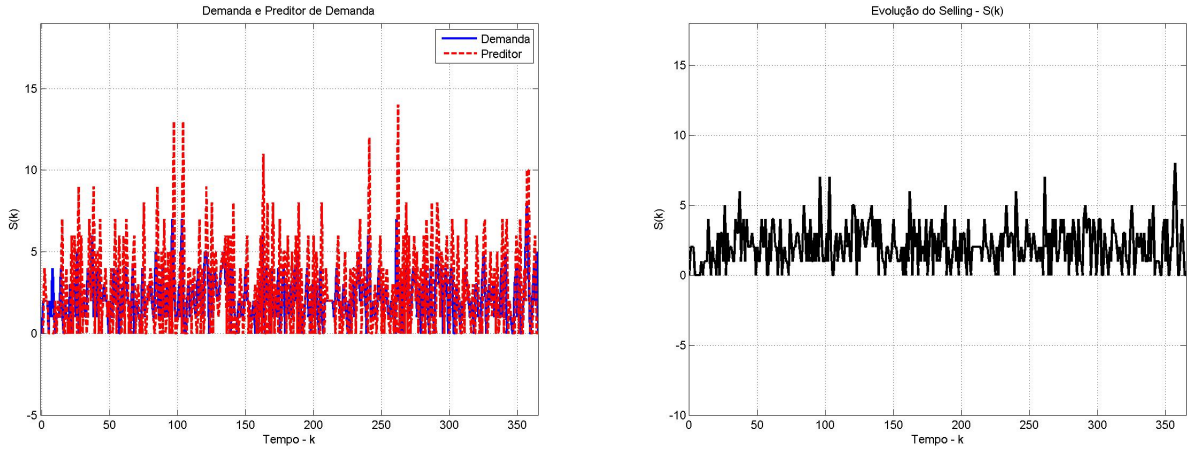

Figura 4.3: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais

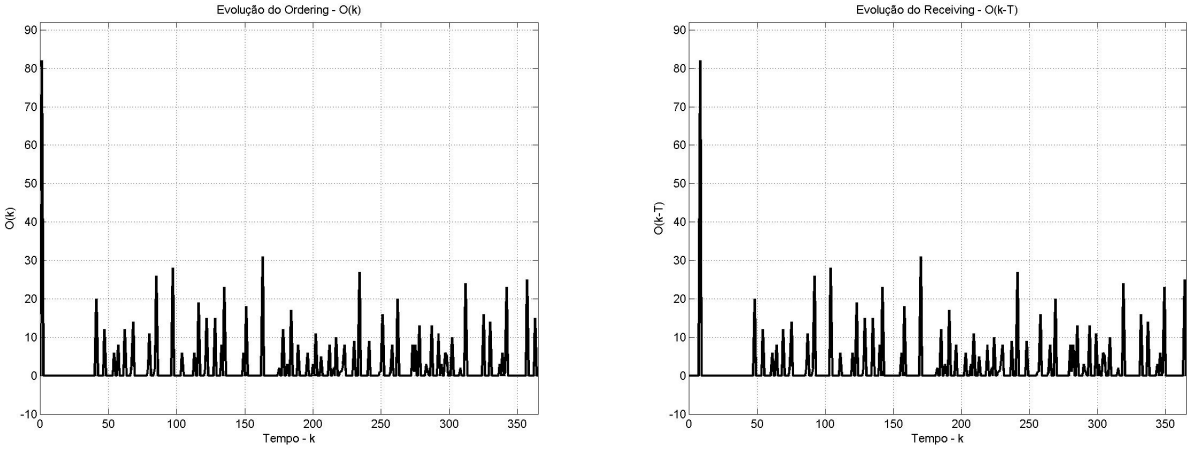

Figura 4.4: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais

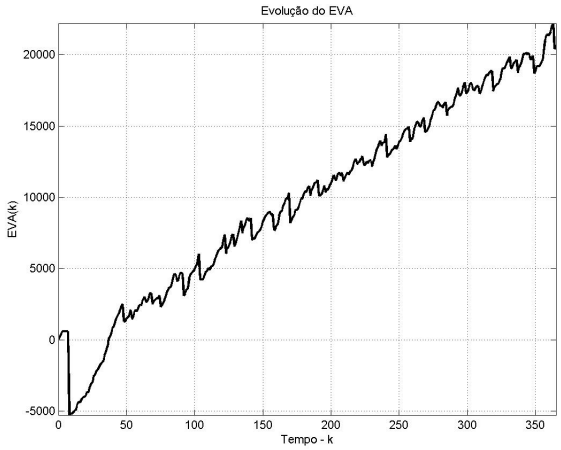

Figura 4.5: Evolução diária do valor do EVA para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em séries temporais

4.2.1.2 Preditor de demanda baseado em suavização exponêncial

Tabela 4.3: Parâmetros obtidos pelo GA em dez simulações para o controlador Garvey-Hannon com demanda estimada por suavização exponencial

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	21372.02	17271.11	19814.05	20054.99	21733.40	**22119.93**	20850.46	20717.45	20740.60	20799.03
I_{sp}	27	65	38	08	23	**16**	35	08	20	18
W_{sp}	2	1	3	131	4	**15**	3	15	6	20
K_{p1}	2.13	6.28	4.16	3.78	2.75	**6.75**	2.16	7.19	4.28	3.53
K_{p2}	1.78	7.31	2.50	0.88	2.34	**4.41**	2.16	6.88	4.38	3.06

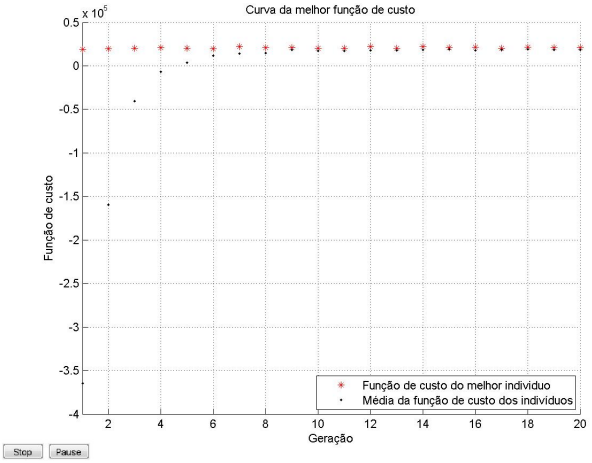

Figura 4.6: Evolução do Algoritmo Genético para o Controlador Garvey-Hannon com demanda estimada por suavização exponencial

Tabela 4.4: Parâmetros e resultados para o controlador Garvey-Hannon com demanda estimada por suavização exponencial

Controlador 1	Valor
I_{sp}	16
W_{sp}	15
K_{p1}	6.75
K_{p2}	4.41
Melhor EVA	19264.62
Média EVA	18046.88
Desvio Padrão EVA	1286.80

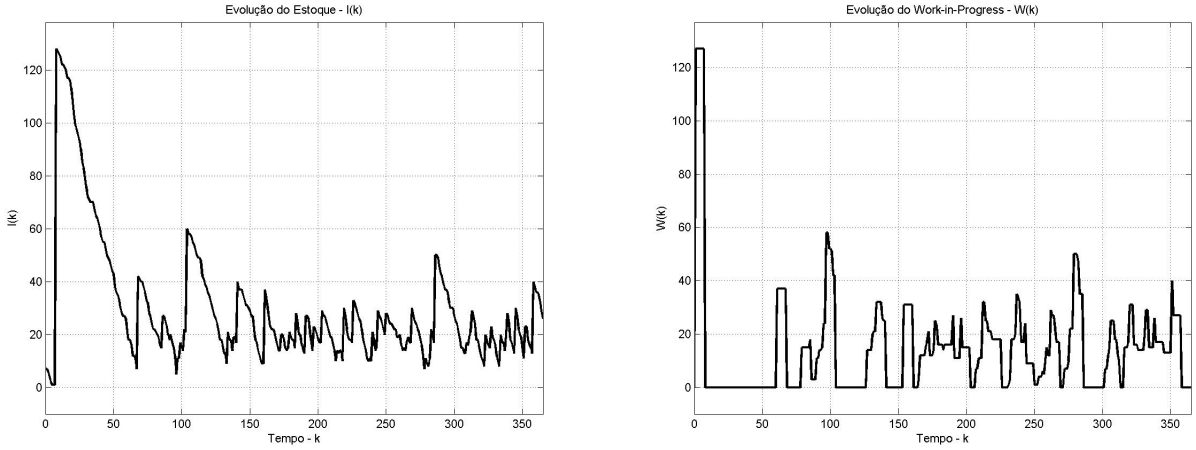

Figura 4.7: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial

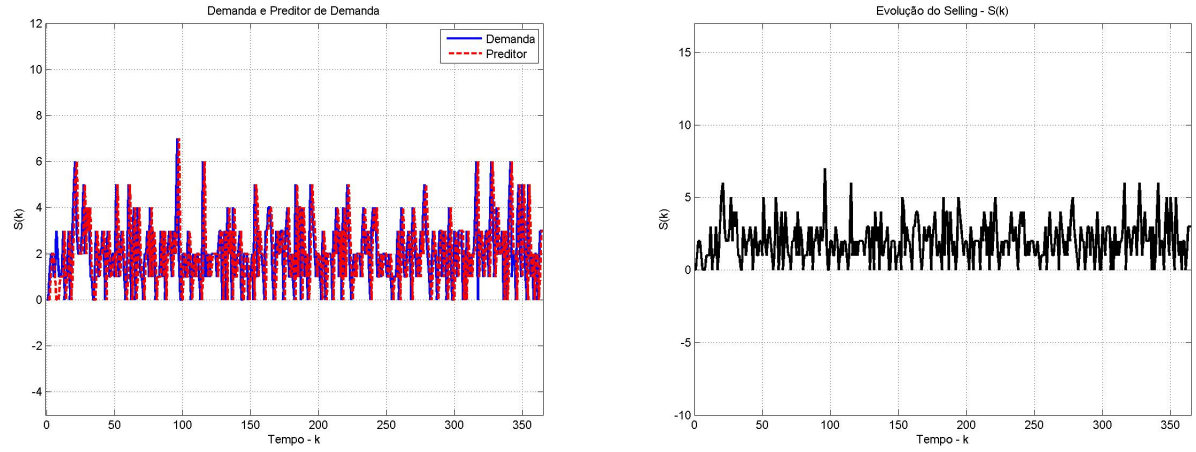

Figura 4.8: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e vendas $S(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial

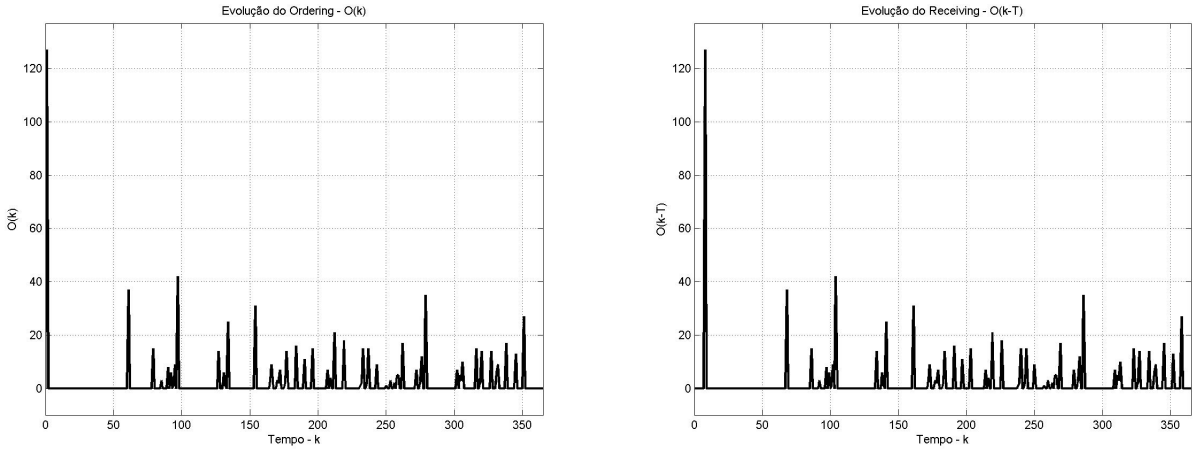

Figura 4.9: Evolução do nível de pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial

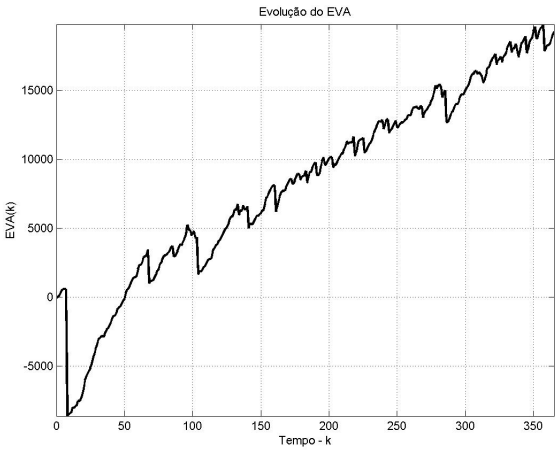

Figura 4.10: Evolução do $EVA(k)$ para o Controlador Garvey-Hannon utilizando preditor de demanda baseado em suavização exponencial

4.2.2 Controlador PID Tosetti

4.2.2.1 Preditor de demanda baseado em séries temporais

Tabela 4.5: Parâmetros obtidos pelo GA em dez simulações para o controlador PID Tosetti com demanda estimada por séries temporais

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	20725.21	20565.12	20805.95	19741.11	19943.42	20699.44	20869.86	21380.42	17075.66	**22064.57**
I_{sp}	22	27	24	35	19	23	26	31	66	**23**
K_p	2.50	4.31	2.44	1.22	5.03	4.00	2.75	2.56	1.16	**2.22**
K_d	0.28	0.03	4.25	0.47	3.28	1.47	1.66	0.66	4.22	**1.16**
K_i	2.28	2.59	6.03	3.03	4.22	6.78	3.78	2.47	7.34	**2.00**

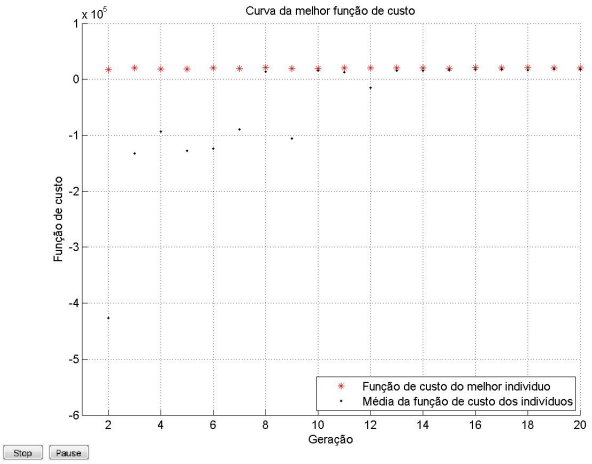

Figura 4.11: Evolução do Algoritmo Genético para o Controlador PID Tosetti com demanda estimada por séries temporais

Tabela 4.6: Parâmetros e resultados para o controlador PID Tosetti com demanda estimada por séries temporais

Controlador 3	Valor
I_{sp}	23
K_p	2.22
K_d	1.16
K_i	2.00
Melhor EVA	20655.09
Média EVA	18300.57
Desvio Padrão EVA	1414.83

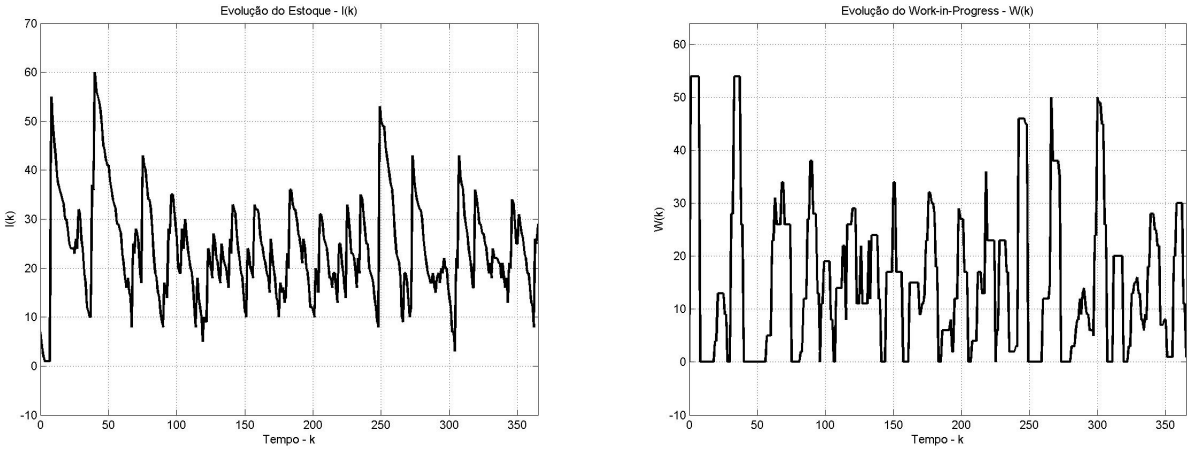

Figura 4.12: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais

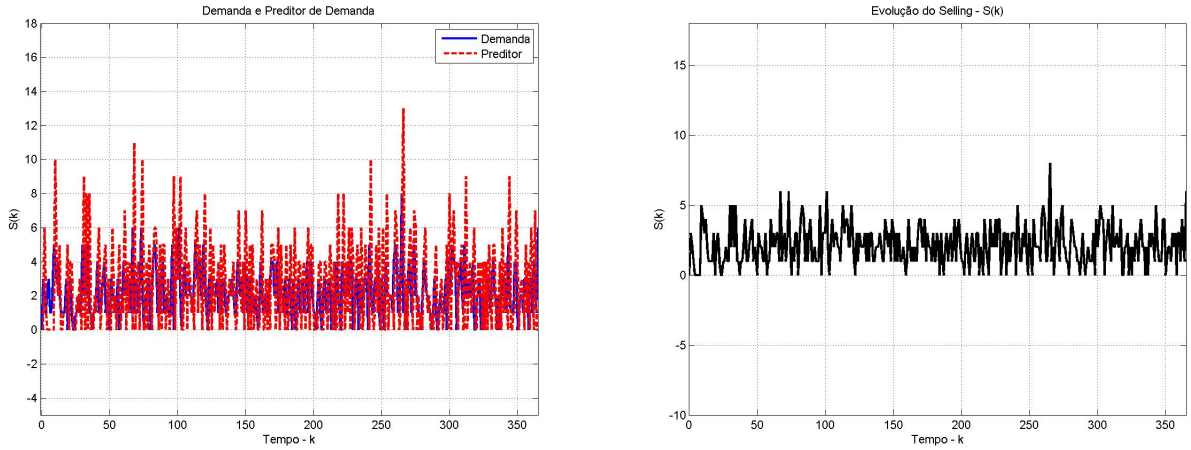

Figura 4.13: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais

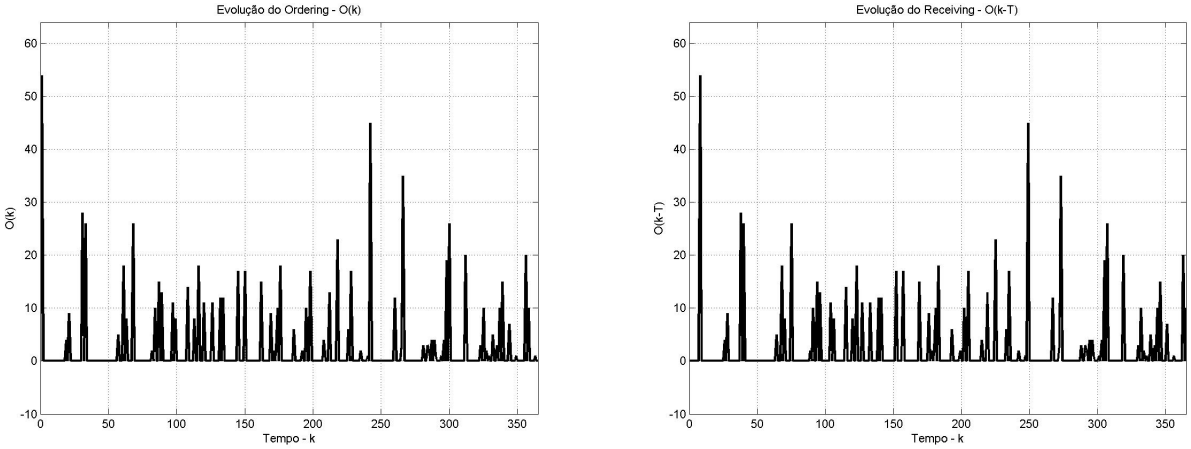

Figura 4.14: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais

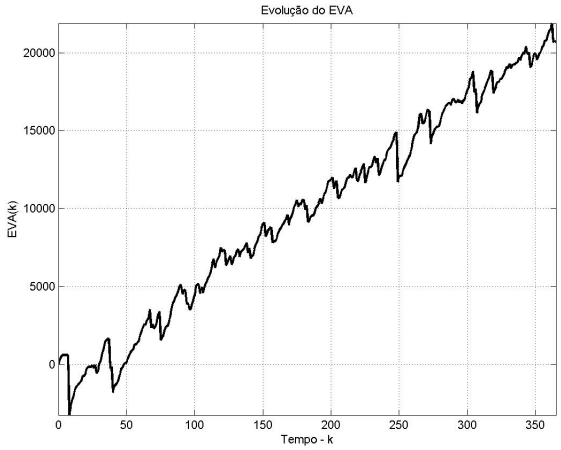

Figura 4.15: Evolução diária do valor do EVA para o Controlador PID Tosetti utilizando preditor de demanda baseado em séries temporais

4.2.2.2 Preditor de demanda baseado em suavização exponencial

Tabela 4.7: Parâmetros obtidos pelo GA em dez simulações para o controlador PID Tosetti com demanda estimada por suavização exponencial

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	20019.26	21361.10	21741.69	21431.10	21029.39	**21949.99**	20995.87	20918.39	21197.12	20523.89
I_{sp}	39	26	28	26	32	**27**	23	37	32	30
K_p	0.09	0.22	0.94	0.59	2.50	**2.90**	0.28	4.19	1.41	1.28
K_d	2.13	2.91	1.19	2.44	2.16	**1.41**	1.34	0.28	0.13	2.00
K_i	7.03	6.97	2.03	3.78	5.06	**7.13**	1.81	7.69	5.25	3.59

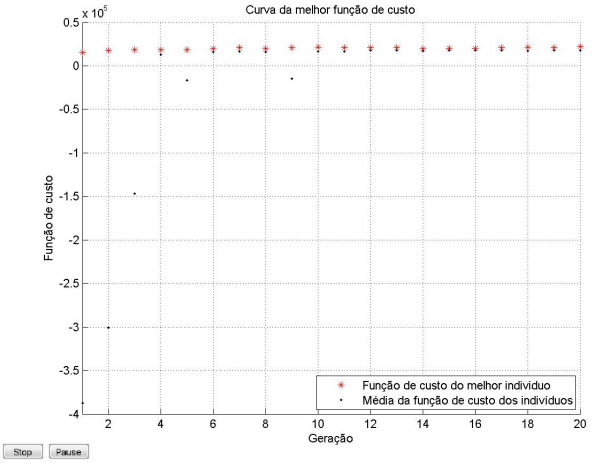

Figura 4.16: Evolução do Algoritmo Genético para o Controlador PID Tosetti com demanda estimada por suavização exponencial

Tabela 4.8: Parâmetros e resultados para o controlador PID Tosetti com demanda estimada por suavização exponencial

Controlador 3	Valor
I_{sp}	27
K_p	2.90
K_d	1.41
K_i	7.13
Melhor EVA	19636.62
Média EVA	18519.83
Desvio Padrão EVA	1285.85

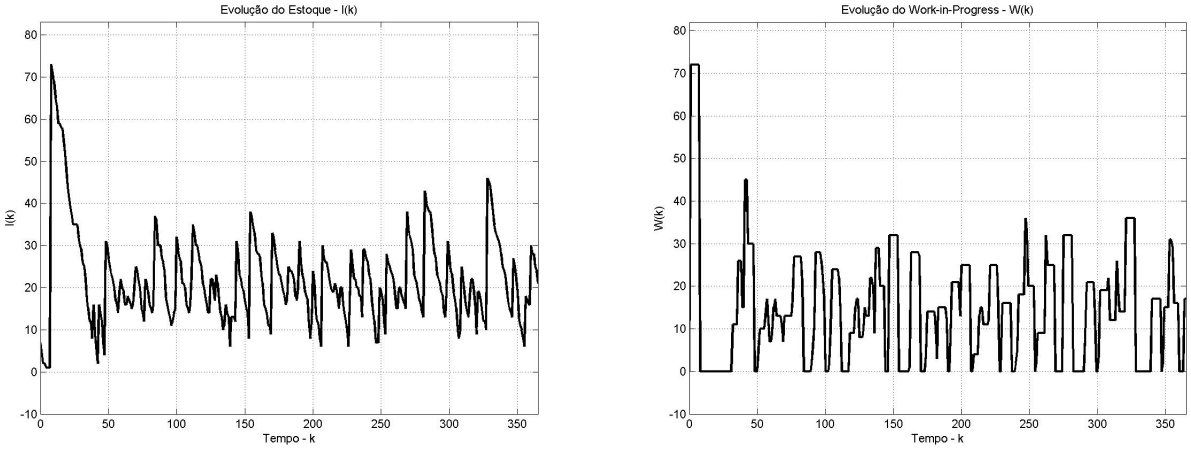

Figura 4.17: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial

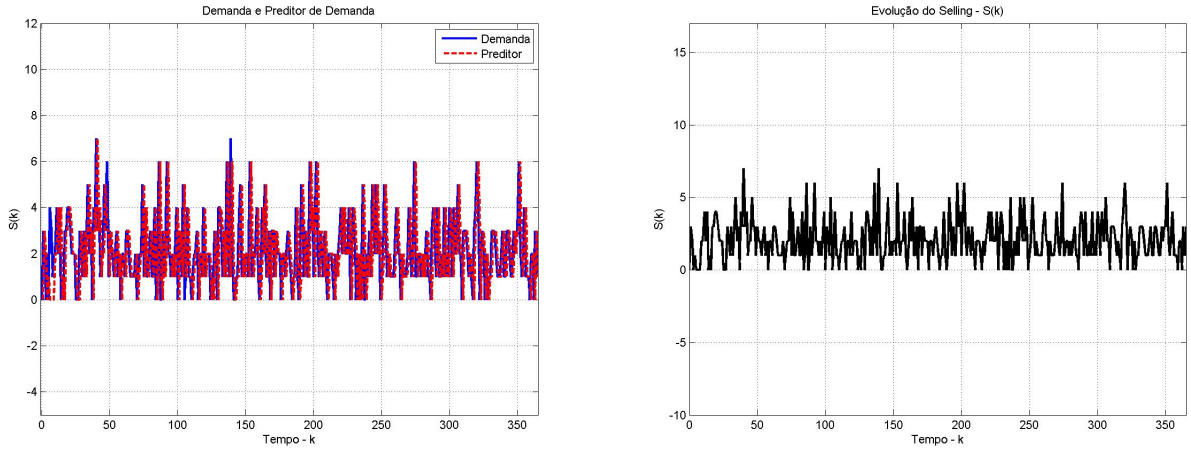

Figura 4.18: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial

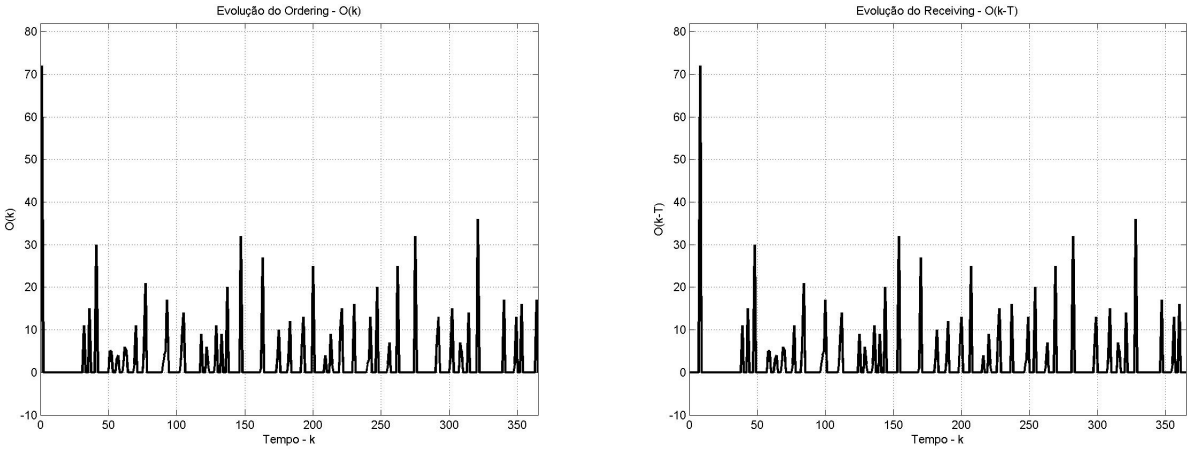

Figura 4.19: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial

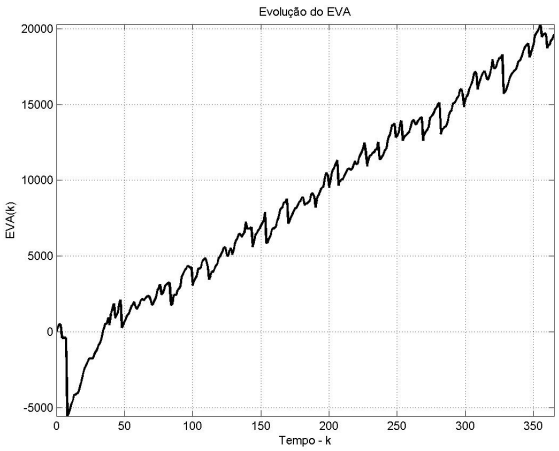

Figura 4.20: Evolução diária do valor do EVA para o Controlador PID Tosetti utilizando preditor de demanda baseado em suavização exponencial

4.2.3 Controlador PID Discreto

4.2.3.1 Preditor de demanda baseado em séries temporais

Tabela 4.9: Parâmetros obtidos pelo GA em dez simulações para o controlador PID Discreto com demanda estimada por séries temporais

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	16292.35	16049.49	17354.46	14281.99	**20880.67**	15233.65	15020.14	19408.43	15637.86	20234.05
I_{sp}	88	86	113	89	**28**	86	86	15	65	34
K_p	2.66	7.53	3.31	7.78	**1.25**	0.16	5.34	4.50	2.69	3.28
K_d	2.06	3.91	3.38	1.00	**0.09**	6.41	0.50	3.91	3.69	0.66
K_i	0.75	0.25	0.94	0.19	**0.00**	0.94	3.94	0.00	1.13	0.00

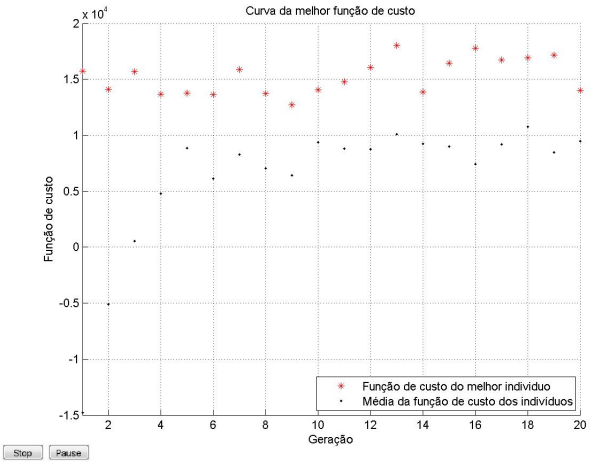

Figura 4.21: Evolução do Algoritmo Genético para o Controlador PID Discreto com demanda estimada por séries temporais

Tabela 4.10: Parâmetros e resultados para o controlador PID Discreto com demanda estimada por séries temporais

Controlador 3	Valor
I_{sp}	28
K_p	1.25
K_d	0.09
K_i	0.00
Melhor EVA	20995.40
Média EVA	19215.65
Desvio Padrão EVA	1135.18

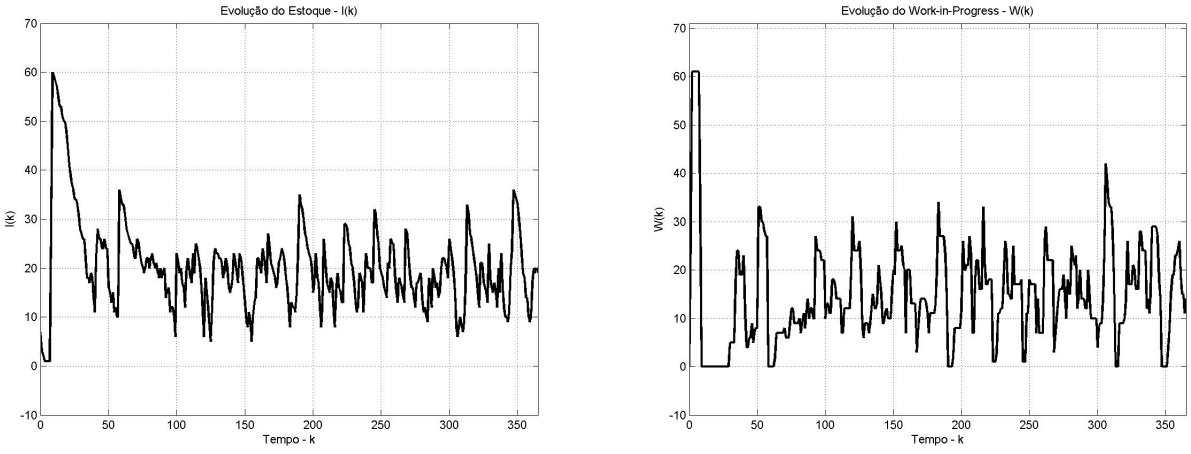

Figura 4.22: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais

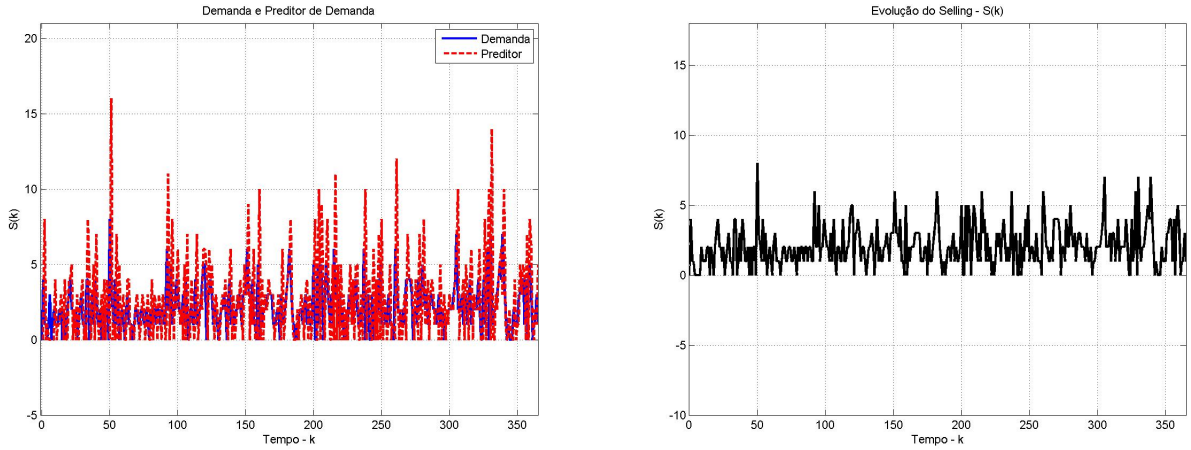

Figura 4.23: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais

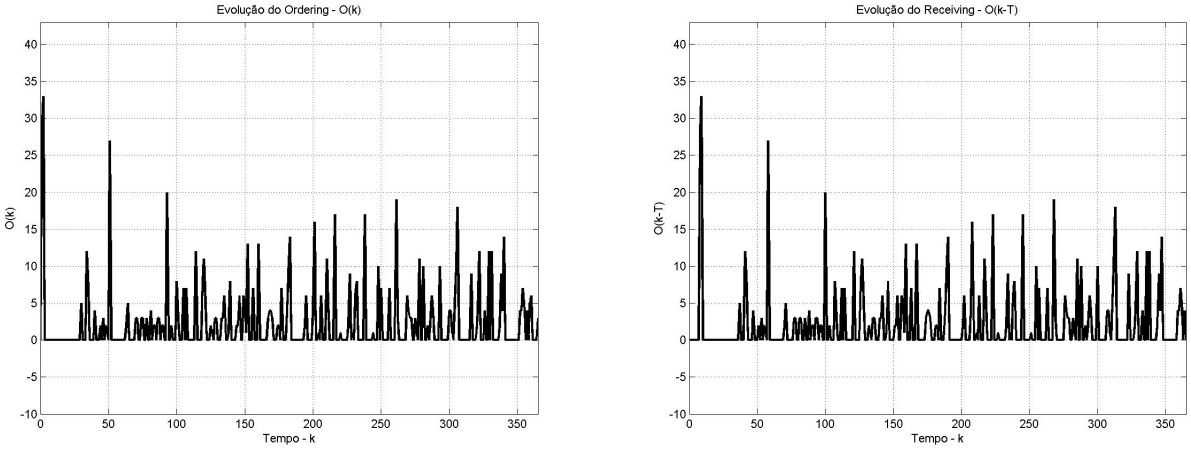

Figura 4.24: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais

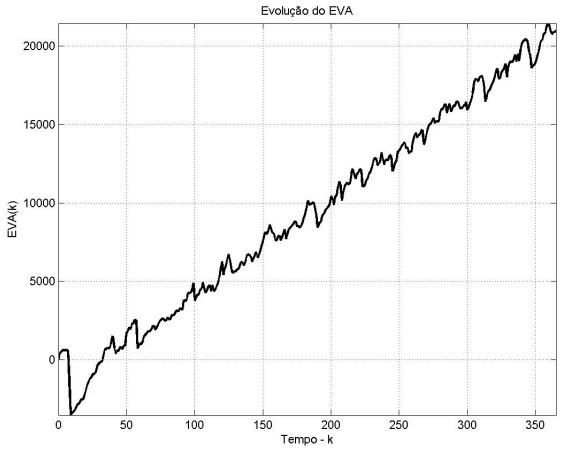

Figura 4.25: Evolução diária do valor do EVA para o Controlador PID Discreto utilizando preditor de demanda baseado em séries temporais

4.2.3.2 Preditor de demanda baseado em suavização exponencial

Tabela 4.11: Parâmetros obtidos pelo GA em dez simulações para o controlador PID Discreto com demanda estimada por suavização exponencial

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	14893.88	16534.09	15885.47	15821.68	17742.74	19156.62	16731.35	**21833.01**	15723.34	16160.06
I_{sp}	80	87	98	88	35	64	75	**51**	96	100
K_p	7.75	7.34	7.59	6.19	1.84	6.59	7.22	**0.38**	7.13	7.81
K_d	7.84	4.41	1.31	6.28	2.00	6.53	2.38	**1.06**	7.63	5.91
K_i	0.06	0.19	0.34	0.75	0.28	0.06	0.03	**0.19**	0.63	0.59

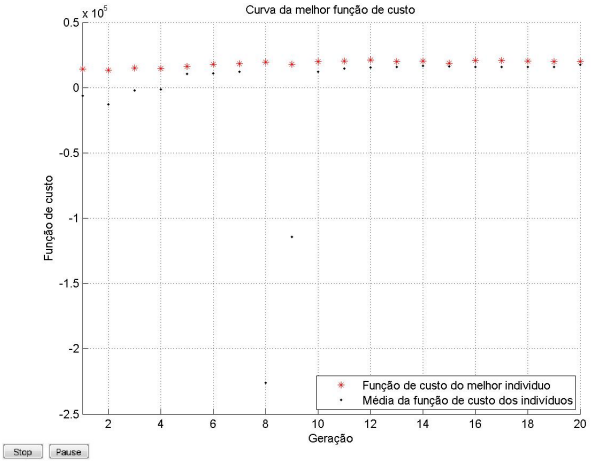

Figura 4.26: Evolução do Algoritmo Genético para o Controlador PID Discreto com demanda estimada por suavização exponencial

Tabela 4.12: Parâmetros e resultados para o controlador PID Discreto com demanda estimada por suavização exponencial

Controlador 3	Valor
I_{sp}	51
K_p	0.38
K_d	1.06
K_i	0.19
Melhor EVA	20151.05
Média EVA	17864.18
Desvio Padrão EVA	1262.47

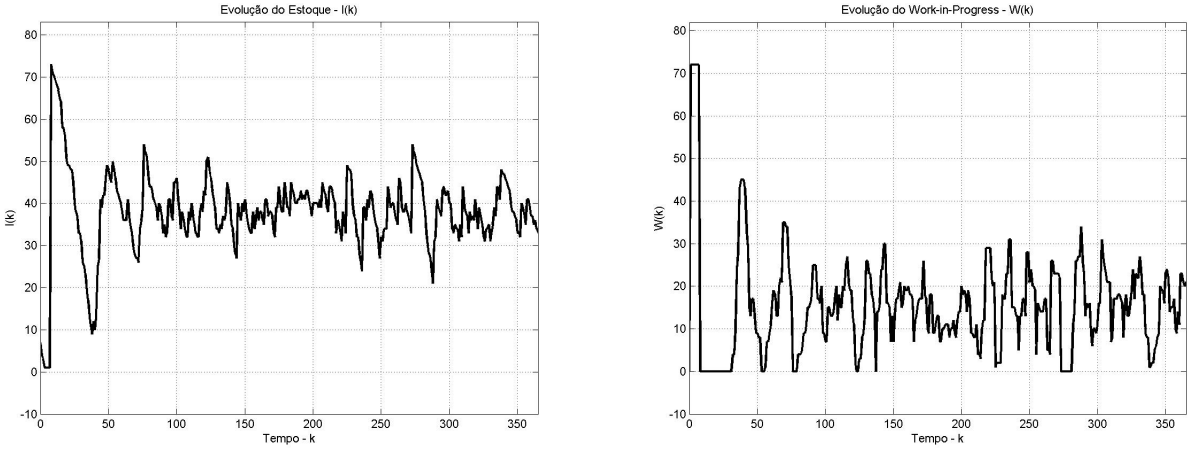

Figura 4.27: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial

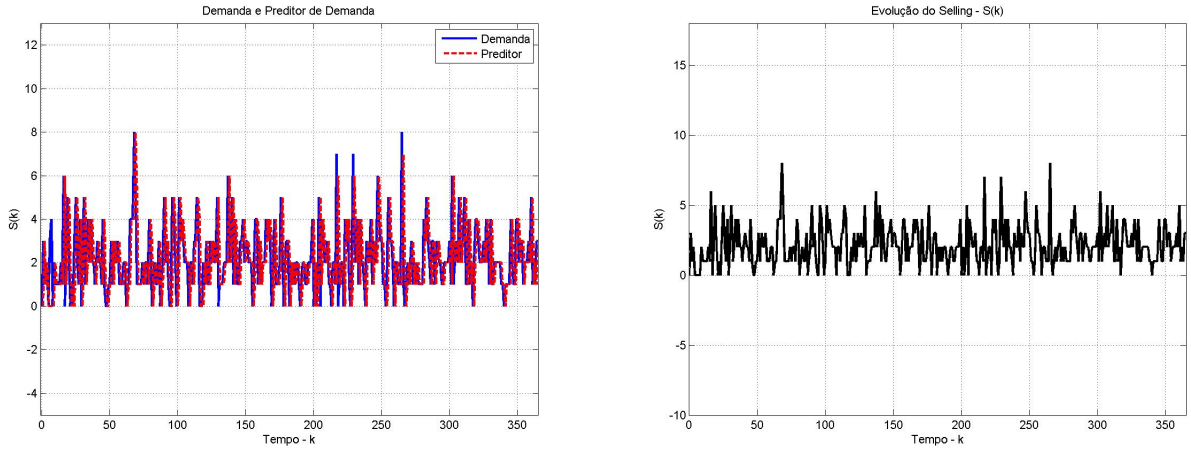

Figura 4.28: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial

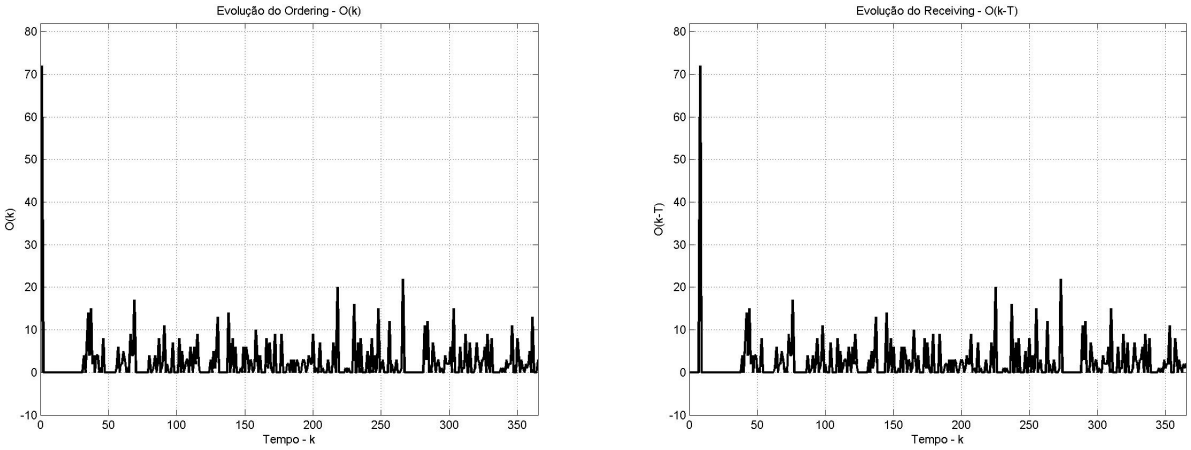

Figura 4.29: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial

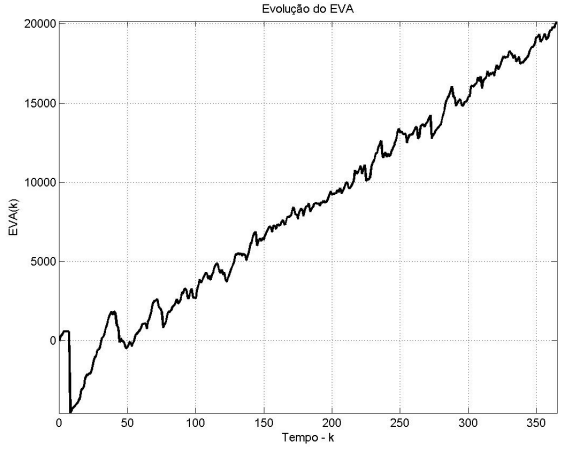

Figura 4.30: Evolução diária do valor do EVA para o Controlador PID Discreto utilizando preditor de demanda baseado em suavização exponencial

4.2.4 Controlador Realimentação de *Receiving*

4.2.4.1 Preditor de demanda baseado em séries temporais

Tabela 4.13: Parâmetros obtidos pelo GA em dez simulações para o controlador Realimentação de *Receiving* com demanda estimada por séries temporais

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	20464.92	19550.95	**21331.58**	16735.20	20899.58	20112.56	19936.16	21101.50	18518.30	20661.71
I_{sp}	5	20	**44**	58	22	1	16	14	14	9
W_{sp}	21	3	**3**	10	5	5	31	6	34	27
R_{sp}	14	25	**1**	129	144	143	6	16	41	5
K_{p1}	6.59	4.84	**1.50**	7.63	3.31	7.56	6.31	5.06	5.84	7.09
K_{p2}	6.25	7.63	**5.03**	7.16	7.41	4.06	3.38	3.72	0.75	6.66
K_{p3}	0.16	3.69	**0.91**	0.06	0.13	1.16	0.19	3.22	1.25	1.06

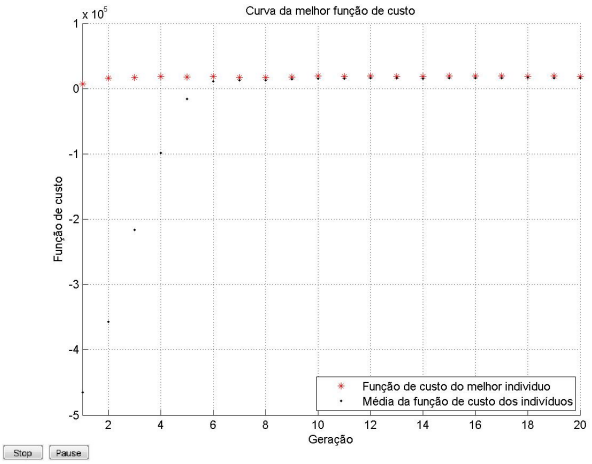

Figura 4.31: Evolução do Algoritmo Genético para o Controlador Realimentação de *Receiving* com demanda estimada por séries temporais

Tabela 4.14: Parâmetros e resultados para o controlador Realimentação de *Receiving* com demanda estimada por séries temporais

Controlador 2	Valor
I_{sp}	44
W_{sp}	3
R_{sp}	1
K_{p1}	1.50
K_{p2}	5.03
K_{p3}	0.91
Melhor EVA	20703.18
Média EVA	17973.41
Desvio Padrão EVA	1615.75

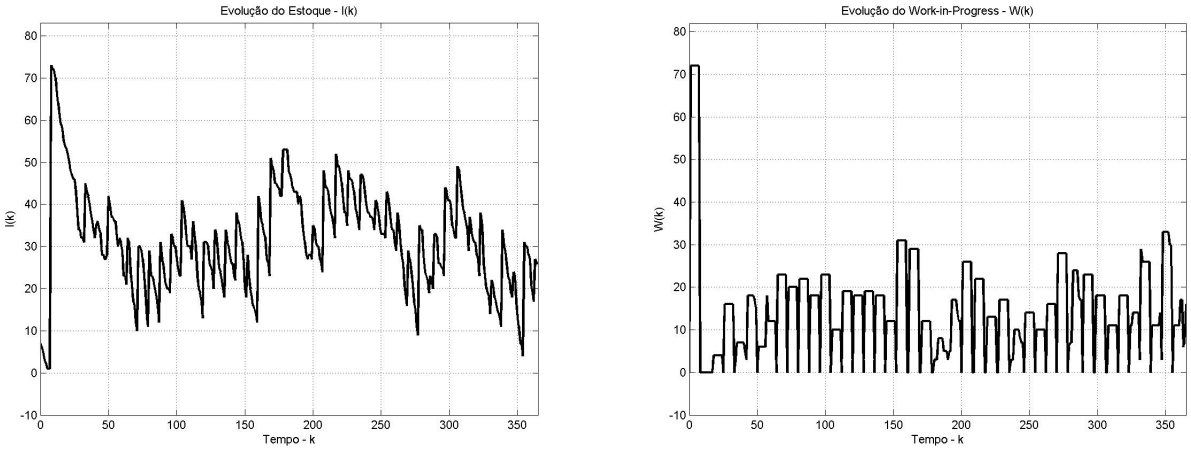

Figura 4.32: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais

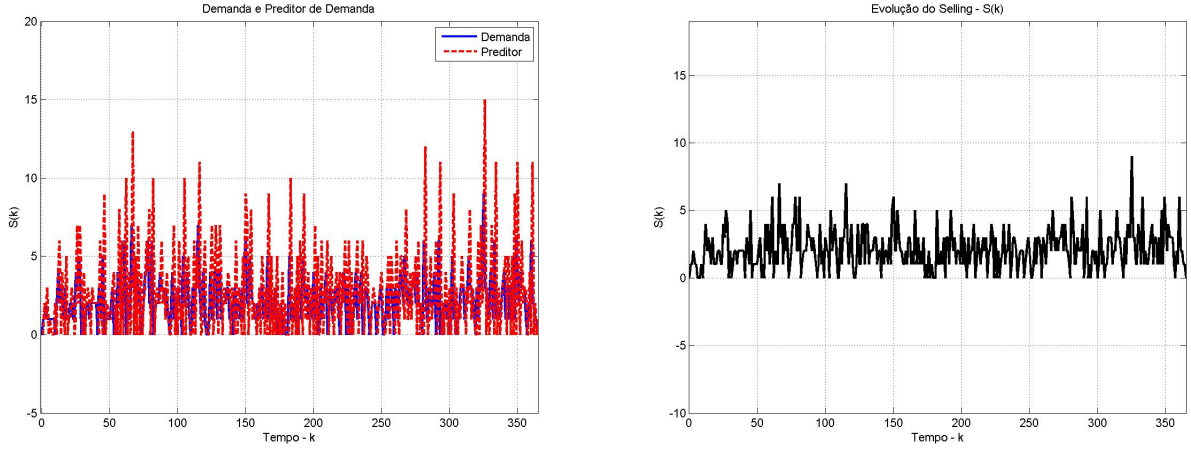

Figura 4.33: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais

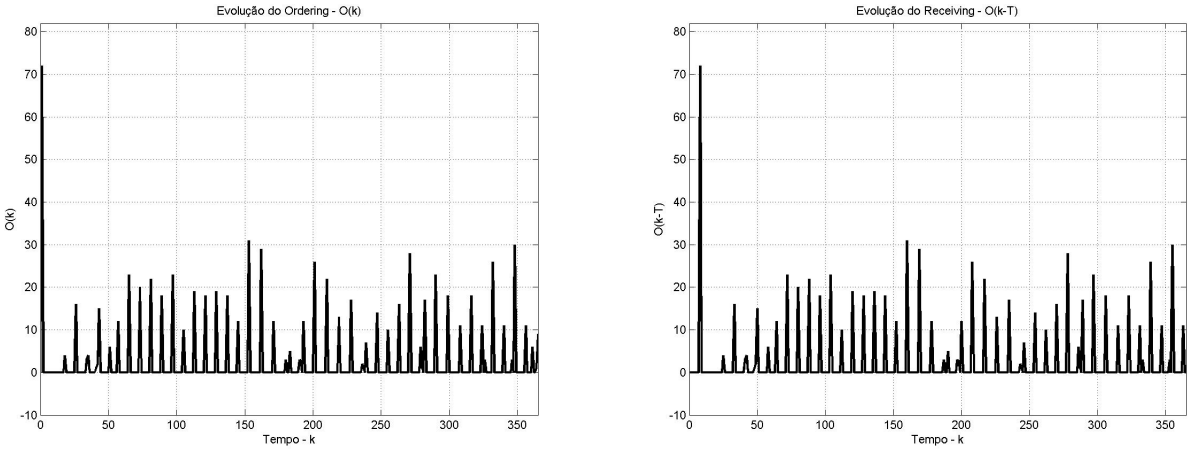

Figura 4.34: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais

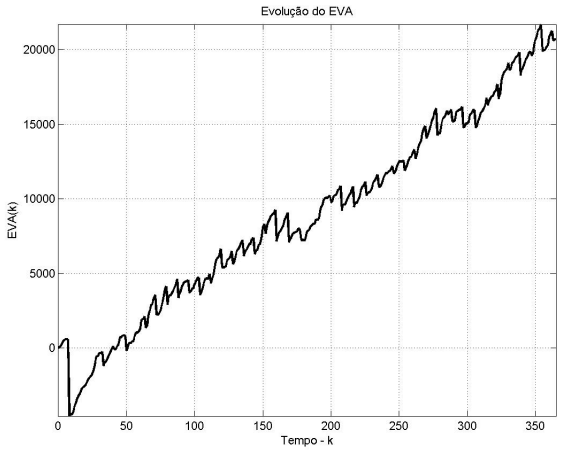

Figura 4.35: Evolução diária do valor do EVA para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em séries temporais

4.2.4.2 Preditor de demanda baseado em suavização exponencial

Tabela 4.15: Parâmetros obtidos pelo GA em dez simulações para o controlador Realimentação de *Receiving* com demanda estimada por suavização exponencial

Parâmetro	1	2	3	4	5	6	7	8	9	10
EVA	20887.91	20775.85	**22598.26**	21126.96	21416.96	21425.91	19671.30	20451.33	21264.51	20165.23
I_{sp}	26	8	**29**	21	22	29	23	25	25	16
W_{sp}	3	29	**0**	10	1	3	18	7	5	2
R_{sp}	0	6	**26**	0	20	23	15	0	3	22
K_{p1}	3.84	6.19	**3.50**	3.84	3.78	2.53	4.31	6.69	2.22	5.06
K_{p2}	4.09	4.06	**3.41**	1.69	5.09	3.91	1.84	5.63	3.09	4.38
K_{p3}	1.66	0.16	**0.75**	7.59	1.22	0.16	0.00	2.31	3.44	3.78

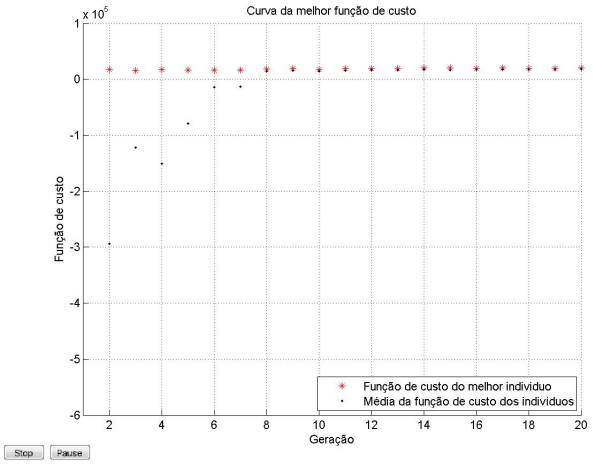

Figura 4.36: Evolução do Algoritmo Genético para o Controlador Realimentação de *Receiving* com demanda estimada por suavização exponencial

Tabela 4.16: Parâmetros e resultados para o controlador Realimentação de *Receiving* com demanda estimada por suavização exponencial

Controlador 2	Valor
I_{sp}	29
W_{sp}	0
R_{sp}	26
K_{p1}	3.50
K_{p2}	3.41
K_{p3}	0.75
Melhor EVA	21010.42
Média EVA	18717.39
Desvio Padrão EVA	2032.58

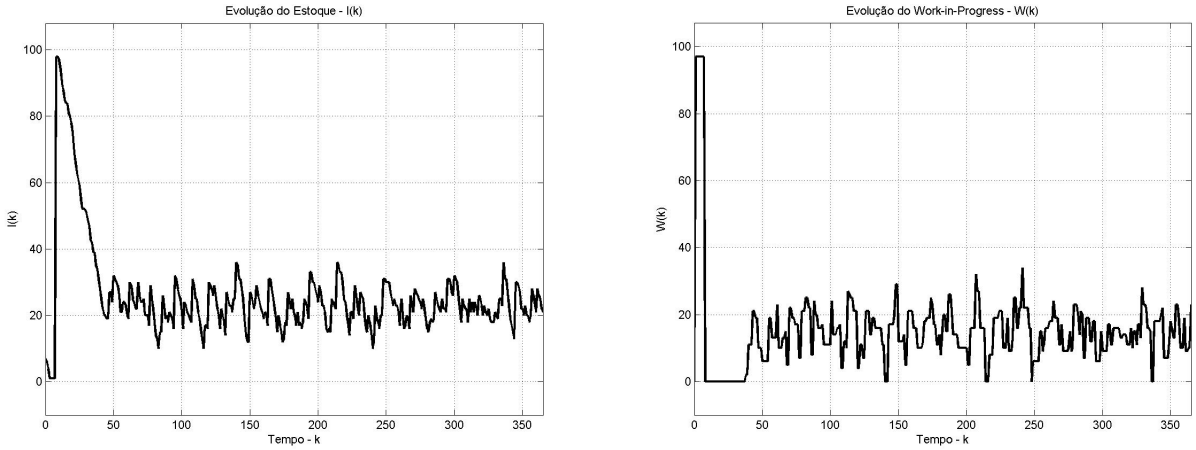

Figura 4.37: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial

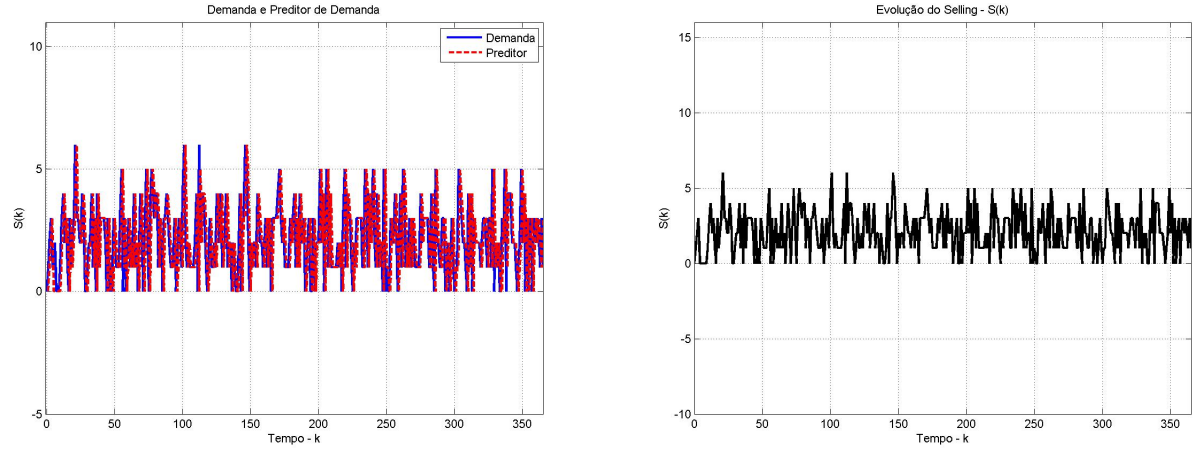

Figura 4.38: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial

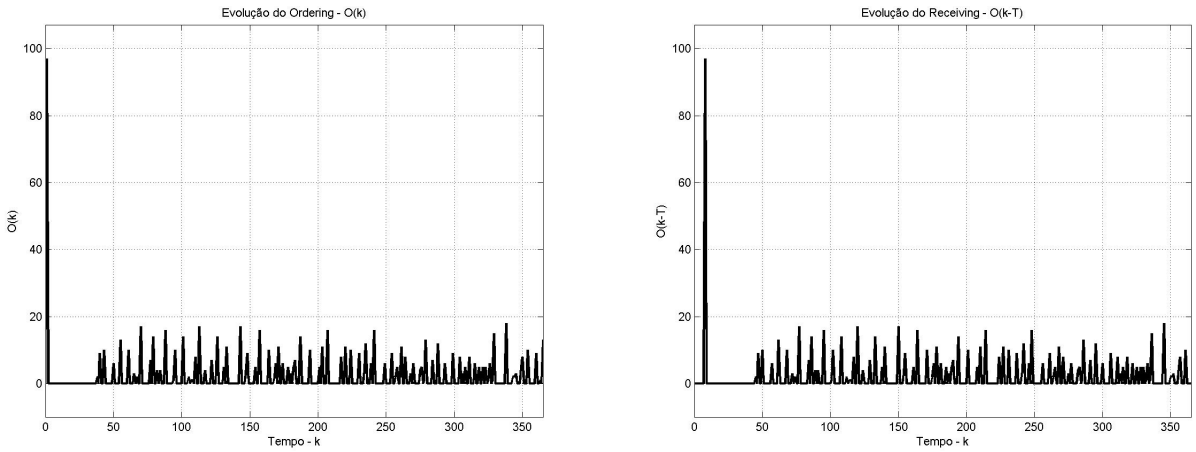

Figura 4.39: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial

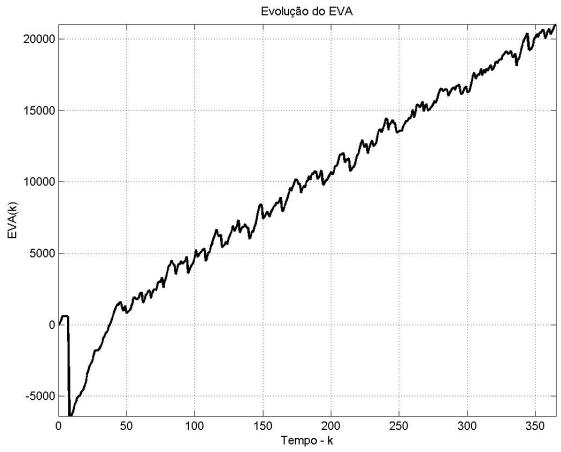

Figura 4.40: Evolução diária do valor do EVA para o Controlador Realimentação de *Receiving* utilizando preditor de demanda baseado em suavização exponencial

4.2.5 Controlador *Fuzzy*

Para o controlador *Fuzzy* não foi possível fazer 10 rodadas do GA. Isso se deveu ao alto tempo requerido para a simulação dessa maneira. Assim, simulou-se apenas uma vez o GA, e, com os parâmetros obtidos, o sistema foi simulado 10 vezes.

Isso, entretanto, não foi prejudicial ao desempenho desse controlador, que encontrou performances superiores aos anteriores, como se vê a seguir.

4.2.5.1 Preditor de demanda baseado em séries temporais

Tabela 4.17: Parâmetros e resultados para o controlador *Fuzzy* com demanda estimada por séries temporais

Controlador 8	Valor
I_{sp}	33.00
W_{sp}	10.00
Melhor EVA	22225.05
Média EVA	20175.70
Desvio Padrão EVA	1575.87

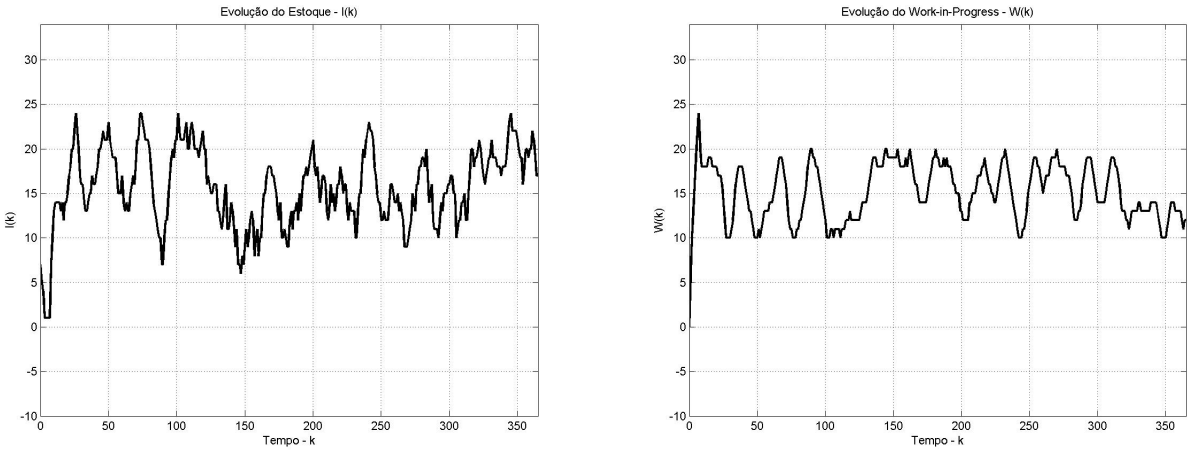

Figura 4.41: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais

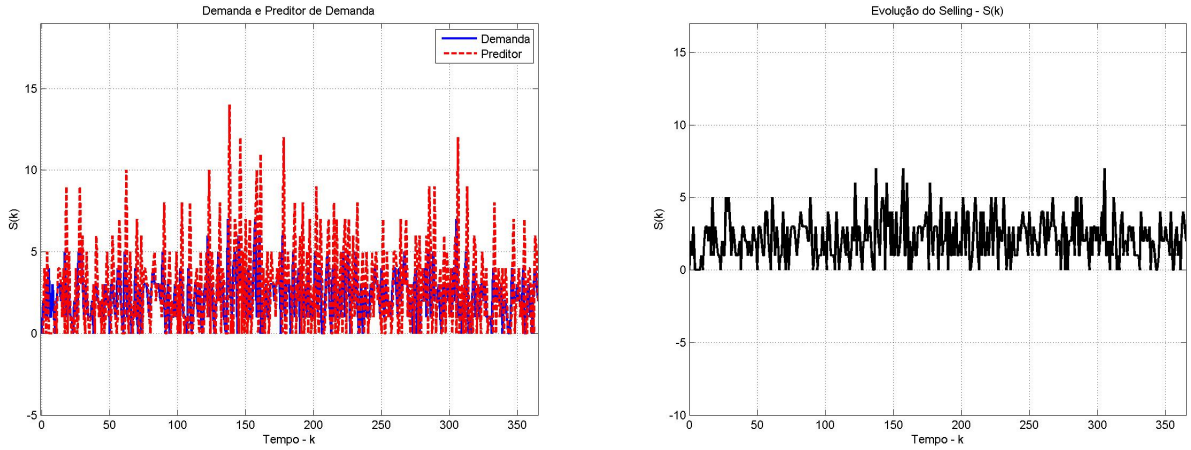

Figura 4.42: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais

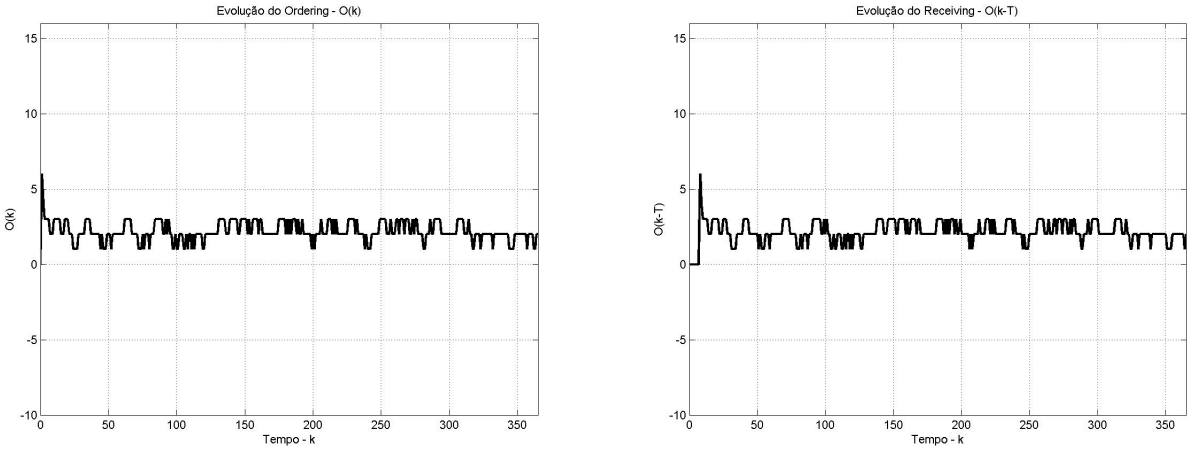

Figura 4.43: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais

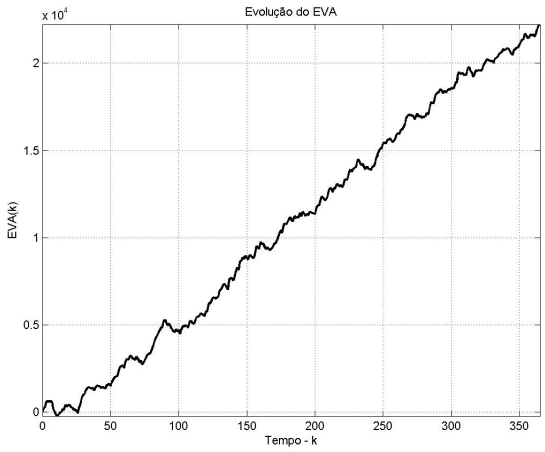

Figura 4.44: Evolução diária do valor do EVA para o Controlador *Fuzzy* utilizando preditor de demanda baseado em séries temporais

4.2.5.2 Preditor de demanda baseado em suavização exponencial

Tabela 4.18: Parâmetros e resultados para o controlador *Fuzzy* com demanda estimada por suavização exponencial

Controlador 8	Valor
I_{sp}	13.00
W_{sp}	16.00
Melhor EVA	21184.26
Média EVA	19829.09
Desvio Padrão EVA	1149.85

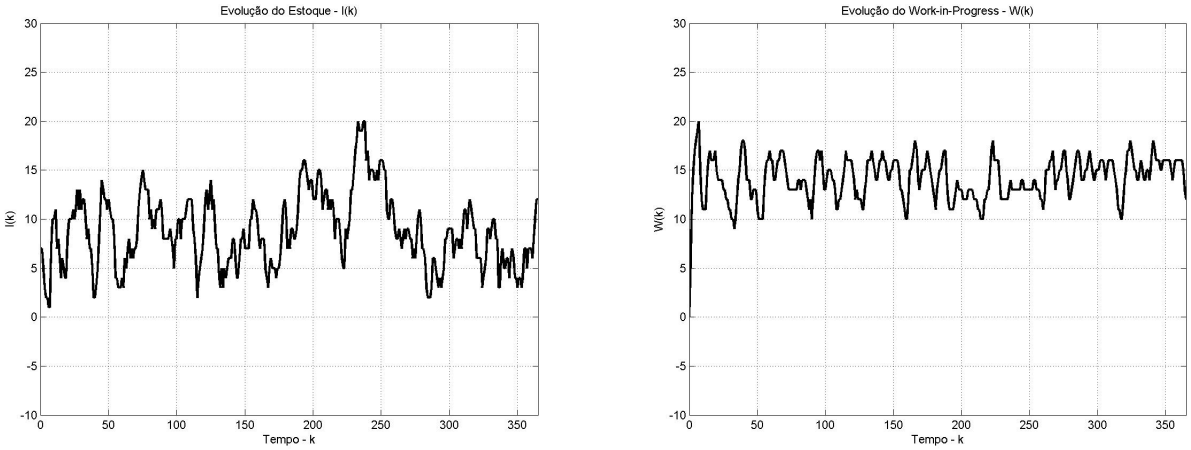

Figura 4.45: Evolução do nível de Estoque $I(k)$ e *Work-In-Progress* $W(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial

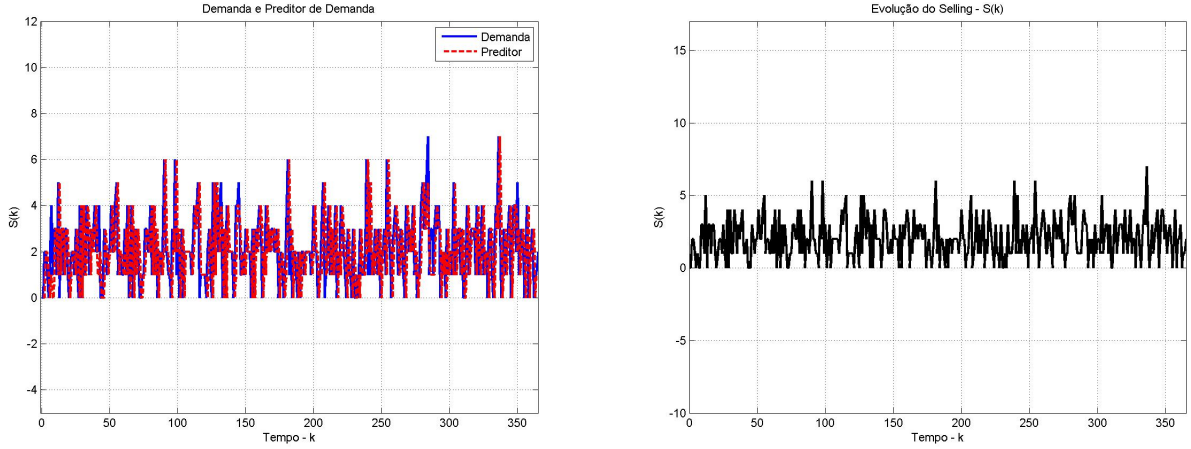

Figura 4.46: Evolução do nível de demanda $D(k)$, do preditor de demanda $\hat{D}(k)$ e das vendas $S(k)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial

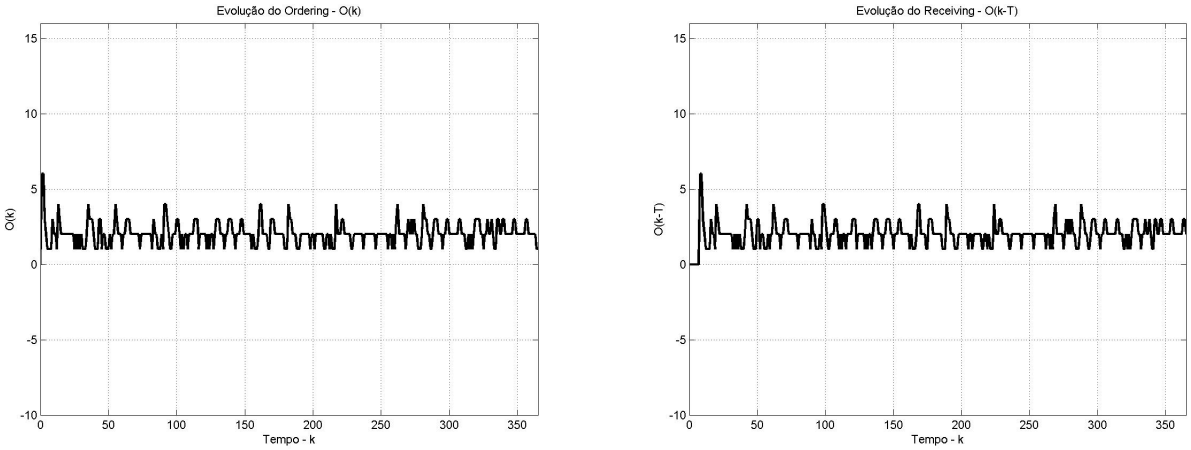

Figura 4.47: Evolução do nível de Pedidos $O(k)$ e $O(k-\tau)$ para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial

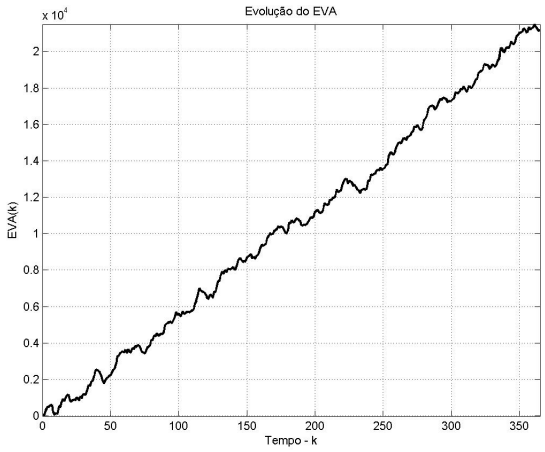

Figura 4.48: Evolução diária do valor do EVA para o Controlador *Fuzzy* utilizando preditor de demanda baseado em suavização exponencial

4.3 Análise dos Resultados

Nota-se, desde o princípio, que o Algoritmo Genético atingiu pontos de EVA máximo semelhantes em todas as estratégias adotadas. Em geral, os valores máximos se aproximam da casa dos $\$21000$. Nesse caso também, houve pouca influência do sistema de predição no valor final do EVA. Além disso, como se pode observar nas Figuras de evolução do algoritmo genético, esse converge rapidamente para os melhores pontos em todas as situações.

Ainda é fundamental ressaltar que o EVA não é sempre monotonicamente crescente, ape-

sar dos resultados apresentados. Esses resultados são fruto de uma boa escolha de parâmetros do controlador pelo GA, mas não são todos os parâmetros dentro o espaço de busca que produzem resultados dessa forma. Como exemplo, se utilizarmos $I_{sp} = 120$, $W_{sp} = 80$, $K_{p1} = 0.1$ e $K_{p2} = 4$ para o controlador Garvey-Hannon e Preditor 1, o EVA final atingirá valores próximos a $ -58000,00. Isso pode ser observado na Figura 4.49

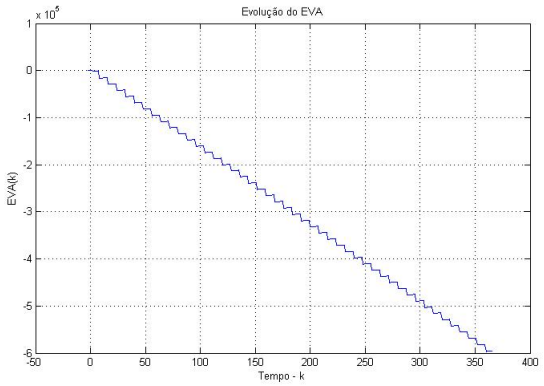

Figura 4.49: Evolução diária do valor do EVA para $I_{sp} = 120$, $W_{sp} = 80$, $K_{p1} = 0.1$ e $K_p2 = 4$

Importante ressaltar que os máximos encontrados derivam de parâmetros bastante distintos entre si. Por exemplo, na tabela 4.3, com o Controlador 1 e Preditor 2, nota-se que existe um ponto de máximo encontrado pelo GA que leva a uma referência para o *Work-in-Progress* em $W_{sp} = 131$. Ao mesmo tempo, é possível encontrar valores da ordem $W_{sp} = 1$ ou $W_{sp} = 4$ com a mesma estratégia. Essa característica do sistema de estoque se repete nas outras estratégias estudadas, mostrando a existência de diversos ótimos locais dispersos pelo espaço de busca.

Analisando o sinal de controle $O(k)$, podemos notar que as estratégias de controle Garvey-Hannon (Controlador 1), Realimentação Receiving (Controlador 4), PID Discreto (Controlador 3) e PID Tosetti (Controlador 2) levam a um pico inicial no volume de pedidos, acarretando todos os EVAs a terem uma queda abrupta no início do ano. Vale ressaltar, também, que, em geral, o Preditor com suavização exponencial (Preditor 2) leva ao sinal de controle a um pico inicial de maior amplitude.

Percebe-se que nos quesitos amplitude e frequência de variação na taxa de pedidos, o Controlador Fuzzy (Controlador 5) possui ótimo desempenho. As Figuras 4.43 e 4.47, com ambos os sistemas de predição, ilustram um sinal de baixa amplitude máxima e com média de

pedidos próxima a dois itens por dia. Esse número é interessante pois é o valor da média da demanda.

Sobre o estoque, é importante ressaltar que não houve rastreamento da referência I_{sp} em nenhum caso. Novamente, o controlador *fuzzy* trouxe vantagens. O nível máximo do estoque, bem como seus picos, estiveram controlados. Para os demais controladores, como era de esperar após analisar o sinal de controle, os picos atingidos pela estratégia com Preditor 2 são de maiores amplitudes. Mesmo não sendo capaz de rastrear o valor da referência, sendo esta mais alta, maior será a média sobre a qual irá variar o sinal de estoque. Pode-se notar esse fenômeno observando o resultado do Controlador 3 (PID Discreto) para os dois sistemas de predição. Com Preditor 1, obteve-se $I_{sp} = 28$, já com Preditor 2, $I_{sp} = 51$. As Figuras 4.22 e 4.27 mostram que, no segundo caso, a curva esteve com sua média mais elevada.

Analisando o sinal $W(k)$ para as diversas estratégias, nota-se que somente o Controlador 5 (*Fuzzy*) leva a um regime estável sem grandes picos ou quedas até zero, contrastando com o comportamento dos outros controladores, nos quais há momentos em que não há nenhuma peça em trânsito. Assim, percebe-se uma tendência por esses controladores a uma política de pedir um volume alto e em períodos mais espaçados. A política *Fuzzy* possui menos sobressaltos.

Interessante notar que a o sinal de controle $O(k)$ para o Controlador 4 (Realimentação de *receiving*) se aproxima de um trem de impulsos (Figura 4.34). Já o sinal $W(k)$ (Figura 4.32) seria um trem de impulsos com base mais espaçada (em 7 dias).

Caso o objetivo do trabalho fosse apenas desenvolver um sistema de estimação de demanda que se aproximasse da demanda real, o preditor 2 poderia ser considerado um sucesso. Em todos os casos, o sinal estimado é muito próximo do sinal original. Curiosamente, isso não levou às estratégias com esse preditor, uma vantagem sobre o outro preditor no que diz respeito ao valor final do EVA. Com ambos os sistemas, obteve-se valores próximos (independentemente do controlador adotado).

As consequências da alta amplitude inicial observada no sinal de controle $O(k)$ é vista na evolução do EVA. Nota-se, por exemplo, na Figura 4.10, que o EVA inicia-se em queda abrupta e são necessários 50 dias até que se atinja o "zero" (prazo para empatar o investimento, ou seja, não há lucros ou prejuízos). A Figura 4.50 compara a evolução do EVA para todos os

controladores, adotando o preditor 1. Constata-se que todos os controladores, exceto o *fuzzy* , levaram o EVA a uma queda inicial. Como o EVA é um modelo com memória, o início negativo prejudica o valor final.

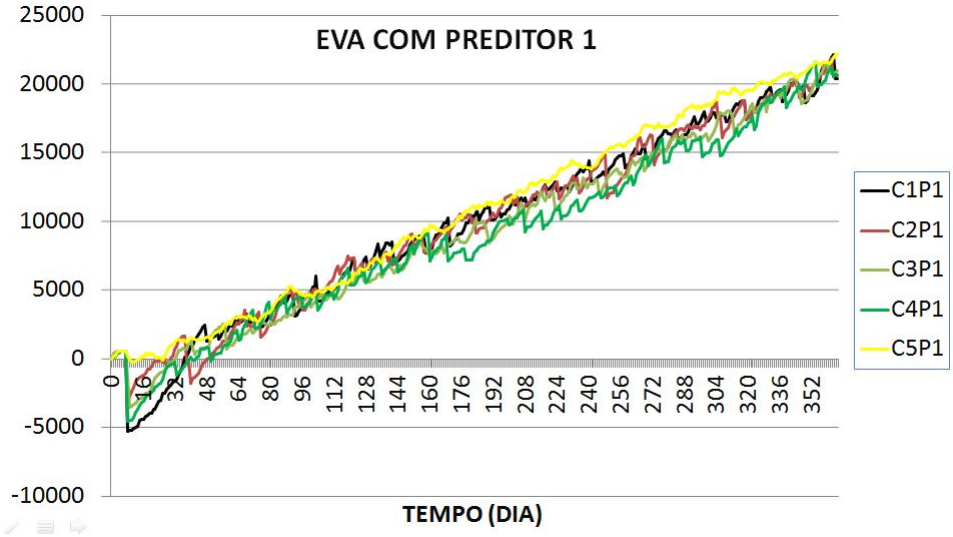

Figura 4.50: Gráfico comparativo da evolução do EVA para todos os controladores utilizando Preditor 1

Ainda podemos analisar comparativamente os valores máximos, médios e o desvio padrão atingidos pelas estratégias ótimas de controle

Tabela 4.19: Melhores valores obtidos pelos controladores ótimos, médias e desvios padrão

Controlador/Preditor	1/1	1/2	2/1	2/2	3/1	3/2	4/1	4/2	5/1	5/2
Melhor EVA	20437.27	19264.62	20655.09	19636.62	20995.40	20151.05	20703.18	21010.42	**22225.05**	21184.26
$\mu(EVA)$	18811.50	18046.88	18300.57	18519.83	19215.65	17864.18	17973.41	18717.39	**20175.70**	19829.09
$\sigma(EVA)$	1799.95	1286.80	1414.83	1285.85	**1135.18**	1262.47	1615.75	2032.58	1575.87	1149.85

Essa tabela mostra comparativamente o resultado obtido pelos controladores. Dela se pode tirar algumas conclusões. A primeira delas é que o Controlador Fuzzy obtém um resultado melhor, mesmo que por uma pequena margem, tanto no melhor resultado quanto na média. É ainda notável que o controlador *Fuzzy* é o único a alcançar valor valor médio acima de $\$20000.00$. Observa-se também que o desvio padrão do Controlador *Fuzzy* está abaixo da média geral.

Ainda se pode observar que os resultados obtidos pelo Controlador PID Discreto apontam um desvio padrão mais reduzido em geral, tendo, inclusive, o menor desvio padrão associado,

para o preditor baseado em séries temporais. Ainda sobre esse método, ele alcança resultados superiores aos demais métodos de controle por realimentação quando utilizado com estimativa de demanda por séries temporais.

Capítulo 5

Conclusão

5.1 Resultados Gerais

As experiências iniciais mostraram que a sintonia manual dos controladores propostos é difícil. A razão é a existência de múltiplos máximos locais e, adicionalmente, o fato de que existem pontos no espaço de busca de parâmetros, muito próximos a estes, com valores de EVA muito mais baixos. Em palavras mais gráficas, pode-se dizer que a paisagem da superfície de busca apresenta muitos picos rodeados de vales profundos, e é exatamente nesse cenário que se lançou mão dos Algoritmos Genéticos.

A sintonia via Algoritmo Genético se mostrou relativamente rápida e robusta. Variações nos parâmetros do AG, como, por exemplo, aumento do tamanho da população, implicava no aumento da exigência computacional, mas não aumentava significativamente o valor ótimo da função objetivo. Assim, optou-se por manter parâmetros do AG que tornaram mais rápida a tarefa de realizar simulações.

O EVA adotado neste trabalho, independentemente da estratégia de controle escolhida, parecia estacionar-se em um patamar constante[1]. Nota-se que o controlador *fuzzy* (Controlador 5) gera um sinal de controle mais suave e contínuo, embora isso não tenha trazido reflexos positivos para o EVA ao final do período. Ao contrário, a influência foi justamente no período inicial do ano. Enquanto os outros controladores geram um sinal de alta amplitude inicial, o controlador *fuzzy* , por utilizar ganhos baixos para sinais de erros pequenos e não utilizar preditor de demanda, não gera este salto inicial no sinal de controle.

[1]Para os valores numéricos adotados nesse trabalho, o valor de EVA neste patamar é de aproximadamente $22000

Outra observação é que o uso do preditor baseado em suavização exponencial, que segue melhor a tendência do sinal de demanda aparentemente, não produz melhores resultados. Entretanto há estudos na literatura que atribuem bastante influência aos preditores na geração do efeito *bullwhip* .

O controlador *fuzzy* possui, como mostra a tabela 4.19, o melhor média para o valor final do EVA para 10 simulações, maior valor máximo e desvio padrão reduzido.

5.2 Trabalhos Futuros

Nesse trabalho, optamos por priorizar o ajuste dos parâmetros do controlador em um dado cenário. Entretanto, nada impede que sejam feitos estudos para vários outros, variando-se, por exemplo, as constantes financeiras e avaliando o desempenho do controlador. Além dessas constantes, outras que podem ser ajustadas são as condições iniciais de estoque e *Work-in-Progress* e o *lead time* referente ao envio. Também pode-se alterar a função de demanda ou ainda utilizar um outro tipo de função objetivo.

É importante frisar que esse trabalho priorizou maximizar o EVA, mas para esse mesmo sistema pode-se ter diversos outros propósitos. Se, em alguma empresa, for importante que se mantenha constante o nível de estoque, pode-se partir para o projeto de um regulador desse nível que seja robusto no sentido de rejeitar perturbações. Se for mais importante que o controlador funcione em diversos cenários de demanda e condições financeiras, o projeto teria, como prioridade principal, o aspecto da robustez aos parâmetros que caracterizam estes.

Podem também ser estudadas outras meta-heurísticas para encontrar a sintonia ótima do controladores, como o Recozimento Simulado (*Simulated Annealing*), as Redes Neurais, entre outros. Também, é possível que se estude o problema utilizando estratégias presentes na teoria clássica de Controle Ótimo. Outro aspecto que pode ser estudado sob perspectivas diferentes é a predição de demanda. Existem diversos métodos de predição que podem ser empregados a esse problema, como os filtros de Kalman.

É interessante também acrescentar, ao simulador desenvolvido, o cálculo do *bullwhip* do sinal de controle, de acordo com as metodologias encontradas em artigos da área. Essa seria mais uma importante informação para analisar o desempenho de uma estratégia de controle.

Bibliografia

[1] Peter Brockwell and Richard Davis. *Time Series: Theory and Methods*. Springer, New York, NY, 2006.

[2] David A Coley. *An introduction to genetic algorithms for scientists and engineers*. World Scientific Publishing, Singapore, 1999.

[3] J Daniel and Chandrasekharan Rajendran. A simulation-based genetic algorithm for inventory optimization in a serial supply chain. *International Transactions in Operational Research*, 12(1):101–127, 2005.

[4] Jeroen Dejonckheere, Stephen M Disney, Marc R Lambrecht, and Denis R Towill. The impact of information enrichment on the bullwhip effect in supply chains: a control engineering perspective. *European Journal of Operational Research*, 153(3):727–750, 2004.

[5] Jeroen Dejonckheere, Stephen Michael Disney, Marc R Lambrecht, and Denis Royston Towill. Measuring and avoiding the bullwhip effect: A control theoretic approach. *European Journal of Operational Research*, 147(3):567–590, 2003.

[6] SM Disney, MM Naim, and DR Towill. Genetic algorithm optimisation of a class of inventory control systems. *International Journal of Production Economics*, 68(3):259–278, 2000.

[7] Stephen M Disney and Denis R Towill. On the bullwhip and inventory variance produced by an ordering policy. *Omega*, 31(3):157–167, 2003.

[8] M. Sam Fadali. *Digital Control Engineering: Analysis and Design*. Elsevier Inc, Amsterdam, 2009.

[9] David E Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Professional, Reading, MA, 1989.

[10] John J Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1):122–128, 1986.

[11] Robert W Grubbström and Joakim Wikner. Inventory trigger control policies developed in terms of control theory. *International Journal of Production Economics*, 45(1):397–406, 1996.

[12] Bruce Hannon and Bernard McGarvey. *Dynamic Modeling For Business Management: An Introduction*. Springer, New York, 2004.

[13] John H Holland. Adaptation in natural and artificial systems, University of Michigan Press. *Ann Arbor, MI*, 1(97):5, 1975.

[14] Felipe Kaiuca Castelo Branco Khoury. Minimização de custos de produção via programação inteira mista: estudo de caso de planejamento de produção de luminárias. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, 2011.

[15] Jacek Kluska. *Analytical Methods in Fuzzy Modeling and Control*. Springer, Berlin, 2009.

[16] Ioan D. Landau and Gianluca Zito. *Digital Control Systems: Design, Identification and Implementation*. Springer, London, 2006.

[17] Kwang H. Lee. *First Course on Fuzzy Theory and Applications*. Springer, Berlin, 2005.

[18] Robert I. Macey and George F. Oster. *Berkeley Madonna Users Guide*. University of California, Berkeley, December 2009.

[19] Robert I. Macey and George F. Oster. Berkeley Madonna: Modeling and analysis of dynamic systems, February 2013.

[20] MathWorks. Global optimization toolbox, February 2013.

[21] Kannan Moudgalya. *Digital Control*. Wiley, Chichester, 2007.

[22] Stuart Chambers e Robert Johnston Nigel Slack. *Administração da Produção*. Springer-Verlag, New York, 2003.

[23] M Ortega and L Lin. Control theory applications to the production–inventory problem: a review. *International Journal of Production Research*, 42(11):2303–2322, 2004.

[24] Kevin Passino and Stephen Yurkovich. *Fuzzy Control*. Addison-Wesley, Menlo Park, CA, 1998.

[25] P Radhakrishnan, VM Prasad, and MR Gopalan. Genetic algorithm based inventory optimization analysis in supply chain management. In *Advance Computing Conference, 2009. IACC 2009. IEEE International*, pages 418–422. IEEE, 2009.

[26] Colin R Reeves and Jonathan E Rowe. *Genetic algorithms-principles and perspectives: a guide to GA theory*, volume 20. Springer, Boston, 2002.

[27] Daniel E Rivera and Michael D Pew. Evaluating pid control for supply chain management: A freshman design project. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 3415–3419. IEEE, 2005.

[28] J. Keith Ord, Rob Hyndman, Anne Koehler and Ralph Snyder. *Forecasting with Exponential Smoothing: The State Space Approach*. Springer, Berlin, 2008.

[29] Timothy J. Ross. *Fuzzy Logic: With Engineering Applications*. Wiley, New Mexico, 2010.

[30] Farzad Sadjadi. Comparison of fitness scaling functions in genetic algorithms with applications to optical processing. In *Optical Science and Technology, the SPIE 49th Annual Meeting*, pages 356–364. International Society for Optics and Photonics, 2004.

[31] J David Schaffer, Richard A Caruana, Larry J Eshelman, and Rajarshi Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proceedings of the third international conference on Genetic algorithms*, pages 51–60. Morgan Kaufmann Publishers Inc., 1989.

[32] William Siler and James J. Buckley. *Fuzzy Expert Systems and Fuzzy Reasoning*. Wiley, Hoboken, NJ, 2005.

[33] ISEE Systems. ithink: Systems thinking business, February 2013.

[34] Santiago Tosetti, D Patiño, Flavio Capraro, and Adrián Gambier. A new inventory level APIOBPCS-based controller. In *American Control Conference, 2008*, pages 2886–2891. IEEE, 2008.

[35] Hélio Flávio Vieira. *Gestão de estoques e operações industriais*. IESDE, Curitiba, 2009.

[36] Li-Xin Wang. *A Course in Fuzzy Systems and Control*. Prentice Hall, Hong Kong, 2003.

[37] Li Zhou and Stephen M Disney. Bullwhip and inventory variance in a closed loop supply chain. *OR Spectrum*, 28(1):127–149, 2006.

[38] Hans-Jungen Zimmermann. *Fuzzy Algorithms for Control*. Kluwer Academic Publishers, Boston, 1999.

Apêndice A

Funcionamento da *Toolbox Optimization Tools*

A caixa de ferramentas *Global Optimization* do MATLAB é um instrumento útil e poderoso. Fornece ao usúario um meio fácil de utilizar métodos de otimização global para problemas com diversos máximos e mínimos. Nos interessa, primordialmente, a técnica *Genetic Algorithm* (Algoritmos Genéticos - GA) que integra o pacote. Essa seção do texto é baseada em tutoriais da ferramenta encontrados em [20]

Como foi dito na seção 2.3, o GA trabalha para solucionar problemas de otimização se baseando em princípios biológicos, repetidamente alterando a população (conjunto de pontos individuais do domínio do problema). Pela sua natureza randômica, é possível aumentar as chances de encontrar um ótimo global. É extremamente vantajoso quando se trata de funções não lineares, não diferenciáveis ou não contínuas.

A caixa de ferramenta do MATLAB permite a utilização de todas as funcionalidades relacionadas ao algoritmo. É possível facilmente escolher qual o método de mutação, o método de selação, como será criada a população inicial (se o usuário irá escolher alguns pontos ou será randômica), os critérios de parada, o tamanho da população e qualquer outro parâmetro de importância.

Existem duas formas clássicas de se trabalhar com a ferramenta: (1) utilizando a interface gráfica *Optimization Tool* ou (2) por linha de comando. Ambas são formas de se obter os mesmo resultados, cabendo ao usuário optar para a mais adequada a cada aplicação.

Importante lembrar que o GA irá buscar pelo ponto que minimize a função objetivo J.

Logo, caso se opte pelo máximo, é necessário minimizar $-J$.

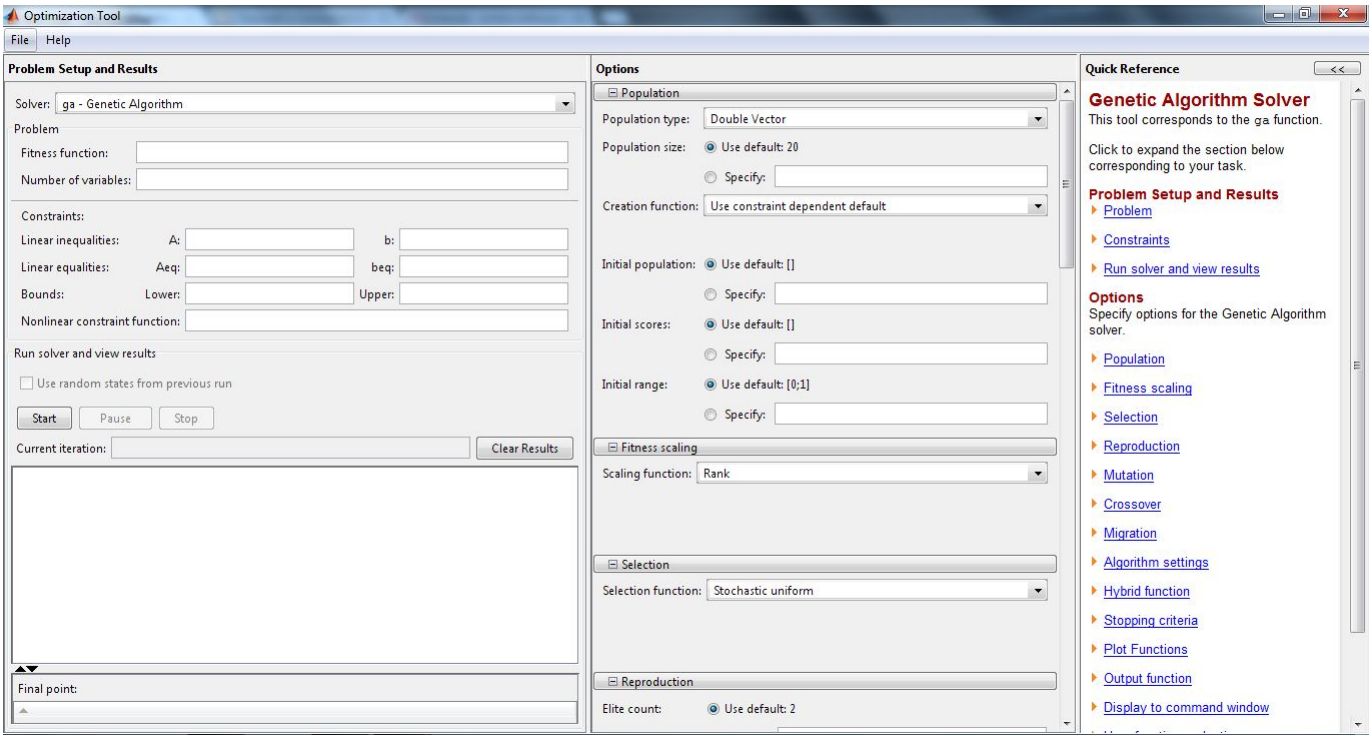

Figura A.1: Tela da Interface Gráfica da ferramenta *Optimization Tool*

Utilizando a linha de commando (ou por arquivos M-File), a seguinte síntaxe é responsável por inicializar o método

```
1 [x fval] = ga(@fitnessfun, nvars, options)
```

onde x é o ponto que minimiza a função objetivo, $@fitnessfun$. O ponto de mínimo é $fval$. O número de variáveis que irá compor um indivíduo da população é $nvars$. Já $options$ discrimina o conjunto de opções possíveis.

Dentre as possíveis opções de configuração se encontram [1]

```
1 options =
2         PopulationType: 'doubleVector'
```

[1] Os valores para as opções são referentes a um exemplo introdutório da própria ferramenta, não tendo relação com os parâmetros utilizados nesse trabalho

```
          PopInitRange: [2x1 double]
        PopulationSize: 20
            EliteCount: 2
      CrossoverFraction: 0.8000
         ParetoFraction: []
     MigrationDirection: 'forward'
      MigrationInterval: 20
      MigrationFraction: 0.2000
            Generations: 100
              TimeLimit: Inf
           FitnessLimit: -Inf
          StallGenLimit: 50
         StallTimeLimit: Inf
                 TolFun: 1.0000e-006
                 TolCon: 1.0000e-006
      InitialPopulation: []
          InitialScores: []
         InitialPenalty: 10
          PenaltyFactor: 100
           PlotInterval: 1
           CreationFcn: @gacreationuniform
      FitnessScalingFcn: @fitscalingrank
           SelectionFcn: @selectionstochunif
           CrossoverFcn: @crossoverscattered
            MutationFcn: {[@mutationgaussian]  [1]  [1]}
     DistanceMeasureFcn: []
              HybridFcn: []
                Display: 'final'
               PlotFcns: []
             OutputFcns: []
             Vectorized: 'off'
            UseParallel: 'never'
```

Não é necessário determinar valores para todas as opções. Caso algum parâmetro não seja determinado, o MATLAB irá utilizar valor padrão.

Explicando as principais opções, temos

- ***PopulationType***: Descreve o tipo de dados utilizado para caracterizar um indivíduo. É possível escolher uma dentre as opções seguintes:
  - ***'bitstring'***: Descrição em bits;
  - ***'custom'***: Tipo de dado personalizado pelo usuário. Não é aplicável a todos os problemas;
  - ***'doubleVector'***: Opção padrão. Indivíduos caracterízados por dados do tipo *double*.

- ***PopInitRange***: Matriz ou vetor que especifica o intervalo permitido aos indivíduos da população inicial;
- ***PopulationSize***: Descreve o tamanho da população. Influência diretamente a qualidade do resultado e a velocidade da simulação;
- ***ElitCount***: Inteiro positivo que descreve quantos indivíduos da geração atual possuem sobrevivência garantida para a próxima geração;
- ***CrossoverFraction***: Fração da população da próxima geração, não incluíndo herdeiros da elite, que é gerado pelo cruzamento de indivíduos. É possível escolher dentre as seguintes opções
- ***ParetoFraction***: Escalar entre zero e um que especifica a fração dos indivíduos a se manterem na primeira frente de Pareto, enquanto a ferramenta seleciona indivíduos de frentes maiores;
- ***MigrationDirection***: Determina a direção da migração;
- ***MigrationInterval***: Inteiro positivo que especifica o número de gerações que se posicionam entre migrações de indivíduos entre subpopulações.
- ***MigrationFraction***: Escalar entre zero e um que especifica a fração de indívuos em cada subpopulação que migra para uma subpopulação diferente;
- ***Generations***: Inteiro positivo que especifica o número máximo de gerações (iterações);
- ***TimeLimit***: Escalar positivo. A ferramenta pára após *TimeLimit* segundos.
- ***FitnessLimit***: Caso a função objetivo atinja esse escalar, a busca cessa;
- ***StallGenLimit***: O algoritmo pára se a mudança da média de pesos relativa na função objetivo para o melhor indivíduo, em *StallGenLimit*, for menor, ou igual, *TolFun*:
- ***StallTimeLimit***: O algoritmo pára se não houver melhorias na função objetivo para o melhor indivíduo em *StallTimeLimit* segundo;

- ***TolFun***: Ver *StallGenLimit*;
- ***TolCon***: Utilizado para determinar a viabilidade em respeito às restrições não lineares. Também $\max(sqrt(eps), sqrt(TolCon))$ determina a viabilidade em respeito às restrições lineares;
- ***InitialPopulation***: Descreve a população inicial para o algortimo. Pode ser randômica;
- ***InitialPenalty***: Valor incial do parâmetro de penalidade;
- ***PenaltyFactor***: parâmetro de atualização da penalidade;
- ***PlotInterval***: Especifica o número de gerações entre iterações seguidas que são representadas graficamente
- ***CreationFcn***: Função utilizada para acriação da população inicial;
- ***FitnessScalingFcn***: Função que determina o escalamento dos valores da função objetivo. Possível escolher dentre
  - ***@fitscalingshiftlinear***: Realiza a escala das pontuações de forma que o valor esperado do melhor indivíduo seja igual a uma constante;
  - ***@fitscalingprop***: Faz o valor esperado proporcional a pontuação do indivíduo;
  - ***@fitscalingtop***: Realiza a escala dos indivíduos com melhores pontuações igualmente;
  - ***@fitscalingrank***: Realiza a escala de pontuações baseados no ranqueamento dos indivíduos, e não em suas pontuações;
- ***SelectionFcn***: Função que seleciona os pais de criaças resultantes de mutação e cruzamento; itemize
- ***@selectionremainder***: Selceiona geradores determinísticamente em uma fração da população. Utiliza o método *roulette* com o restante;
- ***@selectionuniform*** : Seleciona geradores aleatóriamente em uma distribuição uniforme;

- ***@selectionstochunif***: Gera uma linha na qual cada gerador corresponde a seção da linha de comprimento proporcional a sua expectativa. O algortimo se move pela linha em passos de igual dimensão. A cada passo, o algoritmo aloca um pai da seção na qual está;
- ***@selectionroulette***: Simula uma roleta de área igual ao segmento proporcional à expectativa.
- ***@selectiontournament***: Seleciona indivíduos aleatoriamente. Após seleciona o melhor de cada conjunto.

CrossoverFcn: Função utilizada pelo algoritmo para gerar crianças de cruzamentos. Opções possíveis:

- ***@crossoverheuristic***: Heurístico. Cria herdeiros que aleatóriamente se encontram na linha contendo os dois pais, a uma pequena distância do pai com melhor valor para a função objetivo;
- ***@crossoverscattered***: Cria um vetor binário aleatório. Então, quando o elemento do vetor for 1, o método seleciona genes de um pai. Quando for 0, seleciona do outro. Assim, ao final se combina as seleções;
- ***@crossoverintermediate***: Gera herdeiros por uma média de pesos aleatória entre geradores;
- ***@crossoversinglepoint***: Seleciona um ponto aleatório dos indivíduos geradores (o mesmo ponto nos dois), onde ocorrerá a divisão para o cruzamento
- ***@crossovertwopoint*** : Semelhante ao método single point, porém a divisão para o cruzamento ocorre em dois pontos;
- ***@crossoverarithmetic***: Cria herdeiros que são uma média aritimética entre dois pais aleatórios. Permanece no meio da linha entre os dois geradores

MutationFcn: Função que produz crianças mutadas. Opções possíveis:

- ***@mutationuniform***: Mutação uniforme;

- ***@mutationadaptfeasible***: Mutação adaptativa;
- ***@mutationgaussian***: Mutação *gaussiana*.

DistanceMeasureFcn: Função que computa a distância entre indivíduos;

HybridFcn: Função que continua a otimização após o GA ter terminado;

Display: Nível mostrado;

PlotFcns: Opções de gráficos para a visualização da simulação. É possível criar gráficos personalizados;

OutputFcns: Valores de saída do algoritmo GA;

Vectorized: Especifica se a computação da função objetivo é vetorizada;

UseParallel: Computada a função objetivo de uma população em paralelo.